ĐẠI HỌC BÁCH KHOA HÀ NỘI

ĐỒ ÁN TỐT NGHIỆP

Thiết kế hệ thống thử nghiệm (testbed) chuẩn IEEE 802.15.4. Ứng dụng cho giao thức Zigbee

Vũ Minh Đức
Duc.VM212776@sis.hust.edu.vn

Trần Văn Bắc
Bac.TV212700@sis.hust.edu.vn

Ngành Kỹ thuật Điều khiển và Tự động hóa

Giảng viên hướng dẫn: PGS.TS. Nguyễn Quốc Cường

Chữ ký của GVHD

Khoa: Tự động hóa

Trường: Điện - Điện tử

Hà Nội, 07/2025

BỘ GIÁO DỤC & ĐÀO TẠO
ĐH BÁCH KHOA HÀ NỘI

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập – Tự do – Hạnh phúc

NHIỆM VỤ
ĐỒ ÁN TỐT NGHIỆP

Họ và tên sinh viên: Vũ Minh Đức, Trần Văn Bắc
Khóa: K66 Trường: Điện – Điện tử Ngành: KT ĐK & TĐH

1. Tên đề tài

Thiết kế hệ thống thử nghiệm (testbed) chuẩn IEEE 802.15.4. Ứng dụng cho giao thức Zigbee

2. Nội dung đề tài

Thiết kế hệ thống thử nghiệm có chức năng theo dõi và phân tích các thông số lớp PHY và MAC chuẩn IEEE 802.15.4, triển khai hệ thống trên ứng dụng mạng Zigbee thực tế

Vũ Minh Đức : Triển khai phần mềm máy tính trung tâm, các công việc bao gồm

- Phân tích dữ liệu nhận được qua UART từ module log
- Xây dựng các API gửi lệnh điều khiển tới KIT và xử lý phản hồi
- Xây dựng server Flask và giao diện web
- Xây dựng database SQLite lưu trữ dữ liệu
- Tính toán PER và PLR từ các event trong database
- Công việc chung: Xây dựng kịch bản ứng dụng mạng Zibgee, viết hướng dẫn triển khai Testbed

Trần Văn Bắc: Triển khai phần mềm trên KIT, thiết kế phần cứng cho hệ thống, các công việc bao gồm

- Xây dựng module log sử dụng RAIL
- Xây dựng module controller nhận và xử lý lệnh từ máy tính trung tâm
- Xây dựng kịch bản test BER để kiểm thử phần cứng
- Thiết kế mạch cứng cho testbed, tích hợp antena lên mạch
- Công việc chung: Xây dựng kịch bản ứng dụng mạng Zibgee, viết hướng dẫn triển khai Testbed

3. Thời gian giao đề tài: 24/02/2025

4. Thời gian hoàn thành: 14/06/2025

Ngày 20 tháng 06 năm 2025

CÁN BỘ HƯỚNG DẪN

PGS.TS Nguyễn Quốc Cường

LỜI CẢM ƠN

Đây là mục tùy chọn, nên viết phần cảm ơn ngắn gọn, tránh dùng các từ sáo rỗng.

Hà Nội, ngày 05 tháng 09 năm 2022

Sinh viên thực hiện

Nguyễn Văn A

TÓM TẮT ĐỒ ÁN

Tóm tắt nội dung của đồ án tốt nghiệp trong khoảng tối đa 300 chữ. Phần tóm tắt cần nêu được các ý: vấn đề cần thực hiện; phương pháp thực hiện; công cụ sử dụng (phần mềm, phần cứng. . .); kết quả của đồ án có phù hợp với các vấn đề đã đặt ra hay không; tính thực tế của đồ án, định hướng phát triển mở rộng của đồ án (nếu có); các kiến thức và kỹ năng mà sinh viên đã đạt được.

MỤC LỤC

DANH MỤC KÝ HIỆU VÀ CHỮ VIẾT TẮT

HESS	Hybrid Energy Storage System
SC	Super Capacitor
EMS	Energy Management Strategy

DANH MỤC HÌNH VẼ

DANH MỤC BẢNG BIỂU

Bảng cập nhật báo cáo

Bảng 0.1. Bảng cập nhật báo cáo.

Ngày	**Nội dung báo cáo**	**Sửa đổi / ghi chú**
01/10	Chương 1: Giới thiệu	Cập nhật mục tiêu nghiên cứu và phạm vi
08/10	Chương 3: Phương pháp	Thêm mô tả chi tiết về quy trình thực hiện
15/10	Chương 4: Kết quả thực nghiệm	Cập nhật kết quả phân tích dữ liệu mới
22/10	Toàn bộ báo cáo	Chỉnh sửa ngôn ngữ, định dạng

Kế hoạch thực hiện

Bảng 0.2. Bảng kế hoạch dự án.

Tuần	Nhiệm vụ	Yêu cầu cần đạt	Trạng thái
24	Nghiên cứu tài liệu liên quan	Tóm tắt tài liệu, xác định phương pháp	Hoàn thành
25	Thiết kế sơ đồ khối hệ thống	Bản thiết kế sơ đồ khối	Đang thực hiện

Biên bản cuộc họp

Bảng 0.3. Biên bản cuộc họp.

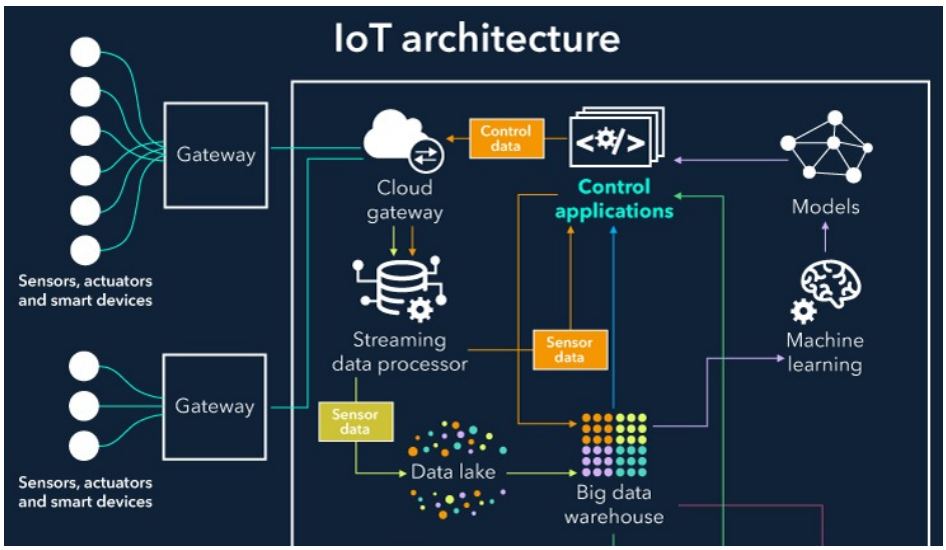
Ngày	Nội dung	Quyết định	Nhiệm vụ tiếp theo
01/10	Thảo luận về thiết kế hệ thống	Sẽ thử với phương pháp A trước	viết báo cáo chương 2

CHƯƠNG 1. GIỚI THIỆU CHUNG

Mở đầu chương

1.1 Bối cảnh và lý do chọn đề tài

Trong thời đại công nghệ số và Cách mạng công nghiệp 4.0, công nghệ truyền thông không dây ngày càng đóng vai trò quan trọng trong việc xây dựng các hệ thống thông minh, tự động hóa và phân tán. Một trong những xu hướng nổi bật là sự phát triển mạnh mẽ của Internet of Things – mạng lưới các thiết bị cảm biến, thu thập dữ liệu từ môi trường và truyền về các trung tâm xử lý để phân tích và đưa ra quyết định. Những ứng dụng IoT có thể được bắt gặp trong hầu hết các lĩnh vực, từ nông nghiệp, giám sát môi trường, nhà thông minh, y tế, quản lý năng lượng, cho đến các hệ thống tự động hóa công nghiệp. Trong đó, các mạng cảm biến không dây là một phần không thể thiếu, đóng vai trò như cánh tay nối dài giúp hệ thống IoT cảm nhận và phản hồi với môi trường.



Hình 1.1. Kiến trúc tổng quát hệ thống IoT

Các mạng cảm biến không dây bao gồm nhiều node cảm biến có khả năng tự tổ chức, hoạt động độc lập bằng pin, tương tác với nhau và truyền dữ liệu không dây đến node điều phối hoặc trung tâm xử lý. Để xây dựng một hệ thống như vậy đòi hỏi phải có những giao thức truyền thông vừa đáng tin cậy, tiêu thụ năng lượng thấp, lại vừa linh hoạt và mở rộng dễ dàng.

Trong số các tiêu chuẩn truyền thông không dây hiện nay, IEEE 802.15.4 đã được công nhận là một nền tảng quan trọng cho các hệ thống mạng cảm biến năng lượng thấp. Đây là tiêu chuẩn được thiết kế riêng cho các hệ thống có yêu cầu cao về tiết kiệm năng lượng, độ trễ thấp, băng thông vừa phải và có khả năng hoạt động trong môi trường hạn chế tài nguyên. Tiêu chuẩn IEEE 802.15.4 quy định chi tiết cho tầng vật lý

(PHY) và tầng điều khiển truy cập môi trường (MAC) – hai tầng cơ sở trong mô hình OSI, đóng vai trò kiểm soát truy cập kênh truyền và truyền dữ liệu hiệu quả. Ngoài ra, chuẩn này còn hỗ trợ các chế độ tiết kiệm năng lượng, cho phép thiết bị ngủ sâu trong thời gian dài mà vẫn duy trì khả năng kết nối. Với đặc điểm như vậy, IEEE 802.15.4 không chỉ được sử dụng như một giao thức độc lập mà còn là nền tảng cho nhiều giao thức lớp cao như Zigbee, WirelessHART, 6LoWPAN, Thread và nhiều hệ thống mạng khác. Nhờ đó, các thiết bị dựa trên chuẩn IEEE 802.15.4 đã và đang được ứng dụng rộng rãi trong thực tế, góp phần tạo nên các mạng cảm biến linh hoạt, hoạt động ổn định trong thời gian dài, đặc biệt ở những nơi không thể cung cấp nguồn điện liên tục.

Tuy nhiên, khi bước vào giai đoạn triển khai thực tế, rất nhiều yếu tố ảnh hưởng đến hiệu năng của các hệ thống truyền thông không dây mà không thể được mô phỏng đầy đủ bằng phần mềm giả lập. Những yếu tố như nhiễu nền, suy hao đường truyền, xung đột kênh, đa đường hay thậm chí các sai số phần cứng đều có thể ảnh hưởng đến các chỉ số như tỷ lệ lỗi bit (BER), tỷ lệ mất gói (PER), thời gian truyền trễ, hoặc chất lượng liên kết (LQI). Chính vì vậy, để nghiên cứu một cách chính xác và đánh giá hiệu năng thực sự của mạng cảm biến không dây, đặc biệt ở tầng MAC và PHY theo chuẩn IEEE 802.15.4, việc xây dựng một hệ thống testbed thực nghiệm là vô cùng cần thiết.

Testbed là một hệ thống kiểm thử thực nghiệm được thiết kế với mục đích cung cấp một môi trường thử nghiệm có kiểm soát, cho phép đánh giá, phân tích và xác thực hoạt động của các hệ thống phần cứng, phần mềm, giao thức truyền thông hoặc các giải pháp công nghệ mới trong điều kiện gần giống thực tế. Không giống như các phương pháp mô phỏng chỉ tái hiện hành vi hệ thống dựa trên các mô hình toán học và giả định lý tưởng, testbed vận hành trên nền tảng các thiết bị vật lý thật và dữ liệu thật, từ đó mang lại kết quả kiểm thử có độ tin cậy và tính thực tiễn cao hơn. Mục tiêu chính của testbed là cho phép người nghiên cứu và phát triển hệ thống quan sát hành vi của hệ thống trong môi trường thực, kiểm tra hiệu suất vận hành, phát hiện các điểm yếu hoặc lỗi phát sinh trong quá trình triển khai, và đồng thời tạo điều kiện để điều chỉnh, tối ưu hóa thiết kế một cách linh hoạt.

Ngoài ra, testbed còn đóng vai trò như một bước trung gian quan trọng trong quá trình chuyển giao công nghệ, từ giai đoạn nghiên cứu lý thuyết sang giai đoạn triển khai thực tế, giúp giảm thiểu rủi ro kỹ thuật và chi phí thử nghiệm quy mô lớn. Với khả năng tái cấu hình linh hoạt, testbed có thể hỗ trợ kiểm thử nhiều kịch bản khác nhau, từ những tình huống đơn giản đến những môi trường phức tạp như mạng không dây đa điểm, hệ thống nhúng thời gian thực, mạng cảm biến quy mô lớn hoặc môi trường công nghiệp có nhiễu cao. Trong bối cảnh phát triển nhanh chóng của các công nghệ như IoT, 5G, trí tuệ nhân tạo và mạng truyền thông thế hệ mới, testbed ngày càng trở thành công cụ không thể thiếu trong các hoạt động nghiên cứu, thử nghiệm và triển khai ứng dụng thực tiễn, đồng thời là nền tảng vững chắc để kiểm chứng các ý tưởng sáng tạo và định hướng cải tiến hệ thống trong tương lai.

Một testbed hoàn chỉnh và hoạt động hiệu quả cần có sự phối hợp của nhiều thành phần phần cứng và phần mềm. Việc thiết kế testbed đòi hỏi sự hiểu biết sâu sắc về kiến trúc hệ thống, mục tiêu kiểm thử cũng như các chỉ tiêu đánh giá. Dưới đây là các thành phần cơ bản và quan trọng của một testbed tiêu chuẩn:

- Thiết bị đầu cuối: Đây là những thiết bị được đưa vào kiểm thử trong testbed, có thể là các cảm biến, thiết bị truyền tín hiệu, node Zigbee, thiết bị IoT, v.v. Mỗi thiết bị thường được lập trình để thực hiện một tác vụ cụ thể như gửi hoặc nhận dữ liệu, đo lường tín hiệu, phản hồi điều khiển hoặc thực thi các thuật toán giao thức. Việc kiểm thử sẽ đánh giá khả năng hoạt động của thiết bị dưới các điều kiện và cấu hình khác nhau.

- Máy tính trung tâm: Máy tính trung tâm đóng vai trò điều phối toàn bộ quá trình kiểm thử trong testbed. Nó chịu trách nhiệm lập trình cho các thiết bị đầu cuối, khởi tạo các kịch bản kiểm thử, theo dõi trạng thái hoạt động và thu thập dữ liệu từ các thiết bị. Ngoài ra, máy tính trung tâm cũng có thể xử lý dữ liệu kiểm thử và hiển thị kết quả cho người dùng theo thời gian thực.

- Phần mềm điều khiển và giám sát: Đây là một phần mềm chạy trên máy tính trung tâm nhằm cung cấp giao diện người dùng để dễ dàng quản lý, cấu hình và điều khiển các phiên kiểm thử. Phần mềm này thường bao gồm các chức năng như tải chương trình cho thiết bị, chọn kịch bản kiểm thử, thu thập dữ liệu đo lường, xuất báo cáo kết quả, và có thể hỗ trợ lập lịch kiểm thử tự động.

- Cơ sở dữ liệu hoặc bộ lưu trữ kết quả: Dữ liệu đo lường và kết quả kiểm thử sẽ được ghi lại và lưu trữ để phân tích sau. Hệ thống lưu trữ này có thể là file log, cơ sở dữ liệu cục bộ hoặc hệ thống lưu trữ từ xa. Các thông số được lưu bao gồm tỷ lệ lỗi, độ trễ truyền, cường độ tín hiệu, công suất tiêu thụ, v.v., giúp đánh giá hiệu suất thiết bị một cách toàn diện.

Những lý do trên đã đặt ra nhu cầu cần xây dựng một nền tảng testbed phục vụ cho việc nghiên cứu và thử nghiệm các hệ thống mạng cảm biến không dây theo chuẩn IEEE 802.15.4.

1.2 Mục tiêu

Mục tiêu chính của đồ án là xây dựng một hệ thống thử nghiệm (testbed) mạng cảm biến không dây sử dụng chuẩn IEEE 802.15.4. Cụ thể sẽ có các kịch bản test hiệu suất truyền nhận của phần cứng và các kịch bản test phần mềm ứng dụng của người dùng. Với việc triển khai testbed, đồ án không chỉ nhằm kiểm thử tính năng truyền thông của mạng mà còn hỗ trợ phân tích chi tiết các thông số ở lớp MAC và PHY – hai lớp then chốt trong mô hình truyền thông OSI.

Về mặt phần cứng, hệ thống testbed sẽ được xây dựng dựa trên mạng Zigbee gồm 4 nút cảm biến sử dụng vi mạch EFR32MG24 của hãng Silicon Labs – một trong những dòng SoC (System-on-Chip) nổi bật với khả năng tích hợp cao, tiết kiệm năng lượng và hỗ trợ truyền thông không dây chuẩn IEEE 802.15.4. Đây là kit phát triển chuyên dụng dành cho các ứng dụng Zigbee, Thread, Bluetooth Low Energy (BLE), và các giao thức mạng không dây công suất thấp khác. EFR32MG24 được tích hợp bộ thu phát RF công suất thấp nhưng có hiệu năng cao, hỗ trợ đầy đủ các chức năng truyền và nhận dữ liệu theo chuẩn IEEE 802.15.4.

Mỗi node trong hệ thống sẽ được cấp nguồn độc lập và giao tiếp với máy tính trung tâm qua UART, cho phép truyền nhận dữ liệu điều khiển cũng như trích xuất các thông tin nội bộ. Các node sẽ được bố trí ở những vị trí cố định trong phòng thí nghiệm, mô phỏng các điều kiện truyền thông thực tế trong môi trường trong nhà, bao gồm cả các yếu tố gây nhiễu như vật cản, khoảng cách truyền và ảnh hưởng của phản xạ tín hiệu. Ngoài ra, trong tương lai có thể mở rộng thêm các cảm biến môi trường (nhiệt độ, độ ẩm, ánh sáng,...) vào các node để mô phỏng đầy đủ hơn một mạng cảm biến thực thụ.

Về mặt phần mềm, máy tính trung tâm đóng vai trò là bộ điều khiển chính và giao diện tương tác với toàn bộ hệ thống testbed. Trên máy tính này, một phần mềm giám sát chuyên dụng sẽ được phát triển nhằm cung cấp đầy đủ các tính năng cần thiết để vận hành và quan sát hoạt động của mạng Zigbee.

Trước hết, phần mềm sẽ đảm nhiệm chức năng nạp chương trình điều khiển cho các node. Cụ thể, người dùng có thể lựa chọn một kịch bản thử nghiệm (ví dụ: gửi gói tin liên tục, gửi theo chu kỳ, truyền gói với độ trễ xác định, v.v.), sau đó phần mềm sẽ tự động biên dịch và nạp firmware tương ứng xuống các thiết bị thông qua giao tiếp UART hoặc thông qua công cụ lập trình tích hợp (như Simplicity Commander của Silicon Labs). Điều này giúp đảm bảo các thiết bị trong mạng đều được cấu hình đúng theo mục tiêu của kịch bản.

Bên cạnh đó, phần mềm cũng cho phép người dùng hiệu chỉnh các thông số truyền thông một cách linh hoạt như: công suất phát (TX Power), khoảng thời gian giữa các gói tin, cơ chế truyền (đơn tuyến, quảng bá), hoặc thậm chí cả các tham số của tầng MAC như thời gian backoff, số lần truyền lại tối đa. Giao diện người dùng sẽ cung cấp các trường nhập liệu và bảng lựa chọn giúp người vận hành dễ dàng cấu hình mà không cần can thiệp vào mã nguồn thiết bị.

Một trong những điểm mạnh của hệ thống phần mềm là khả năng theo dõi và ghi nhận các thông số hoạt động ở lớp MAC và PHY trong thời gian thực. Phần mềm sẽ liên tục thu thập dữ liệu từ các node bao gồm: số lượng gói tin đã truyền và nhận, tỷ lệ lỗi bit (BER), tỷ lệ lỗi gói (PER), cường độ tín hiệu nhận được (RSSI), độ nhiễu tín hiệu (SNR), số lần truyền lại, thời gian trung bình truy cập kênh,... Các thông tin này

sẽ được hiển thị dưới dạng biểu đồ, bảng thống kê và nhật ký hoạt động, giúp người dùng dễ dàng đánh giá hiệu năng mạng trong từng kịch bản thử nghiệm cụ thể.

1.3 Phạm vi nghiên cứu

Đề tài tập trung vào việc xây dựng và triển khai một hệ thống testbed nhỏ phục vụ mục đích kiểm thử và đánh giá hiệu năng truyền thông trong mạng cảm biến không dây sử dụng giao thức Zigbee. Phạm vi nghiên cứu chủ yếu xoay quanh việc thiết kế, lập trình và đo lường các thông số truyền thông ở cấp độ thấp, nhằm cung cấp cái nhìn thực nghiệm về hoạt động của hệ thống mạng cảm biến.

Trong khuôn khổ này, nghiên cứu chỉ sử dụng chuẩn truyền thông IEEE 802.15.4 kết hợp với giao thức Zigbee, phù hợp với các ứng dụng đòi hỏi tiêu thụ năng lượng thấp, phạm vi truyền ngắn và yêu cầu độ tin cậy cao. Các lớp mạng cao hơn như lớp ứng dụng hoặc bảo mật Zigbee nâng cao sẽ không được đề cập chi tiết trong phạm vi đề tài.

Phần mềm và phần cứng trong hệ thống testbed được thiết kế để thu thập và theo dõi các chỉ số kỹ thuật ở tầng vật lý và tầng liên kết dữ liệu, bao gồm: RSSI (cường độ tín hiệu nhận được), LQI (chỉ số chất lượng liên kết) BER (tỉ lệ lỗi bit), PER (tỉ lệ lỗi gói tin), độ trễ truyền và mức tiêu thụ năng lượng. Những thông số này sẽ được ghi nhận và hiển thị thông qua một phần mềm điều khiển trung tâm chạy trên máy tính.

Về phần cứng, hệ thống sử dụng các mô-đun EFR32MG24 của Silicon Labs làm node cảm biến. Mỗi node được kết nối trực tiếp với máy tính trung tâm qua giao tiếp UART, cho phép truyền nhận dữ liệu và cập nhật firmware kiểm thử. Tổng cộng có 5 node cảm biến, hoạt động độc lập hoặc phối hợp theo các cấu trúc mạng hình cây đơn giản cùng với 1 máy tính trung tâm đóng vai trò điều phối.

1.4 Cấu trúc báo cáo

Báo cáo được chia thành 4 chương chính, trình bày theo trình tự từ lý do chọn đề tài đến kết quả thực nghiệm, cụ thể như sau:

- Chương 1 – Bối cảnh và lý do chọn đề tài: Trình bày tổng quan về nhu cầu kiểm thử mạng trong thực tế, lý do lựa chọn đề tài, mục tiêu và phạm vi nghiên cứu của đồ án.

- Chương 2 – Cơ sở lý thuyết: Giới thiệu tổng quan về công nghệ Zigbee, giao thức IEEE 802.15.4, các chỉ số đánh giá hiệu năng như RSSI, LQI, BER, PER, cùng với phần cứng sử dụng và các tài liệu liên quan.

- Chương 3 – Phương pháp luận: Mô tả cách thiết kế hệ thống kiểm thử thực tế với các node Zigbee kết nối UART đến máy tính trung tâm, quy trình gửi lệnh – thu dữ liệu – flash chương trình, và cách xây dựng các kịch bản kiểm thử.

- Chương 4 – Kết quả thực nghiệm: Trình bày kết quả đo lường các chỉ số mạng trong các tình huống khác nhau, phân tích dữ liệu thu được, đánh giá hiệu quả của hệ thống và đề xuất hướng phát triển trong tương lai.

1.5 Kết luận chương

Chương 1 đã trình bày bối cảnh nghiên cứu, nhu cầu kiểm thử mạng Zigbee trong các ứng dụng thực tế, cũng như lý do chọn đề tài. Qua đó, đồ án xác định rõ mục tiêu xây dựng một hệ thống kiểm thử mạng Zigbee có khả năng đo lường các chỉ số ở lớp vật lý và MAC. Nội dung chương cũng giới hạn phạm vi nghiên cứu phù hợp với điều kiện phần cứng và thời gian thực hiện, làm cơ sở cho các chương tiếp theo về lý thuyết, phương pháp và thực nghiệm.

Kết luận chương

CHƯƠNG 2. CƠ SỞ LÝ THUYẾT

Mở đầu chương

2.1 Chuẩn truyền thông IEEE 802.15.4

2.1.1 Giới thiệu chung về IEEE 802.15.4

Chuẩn IEEE 802.15.4 là một tiêu chuẩn kỹ thuật được phát triển bởi IEEE (Institute of Electrical and Electronics Engineers) nhằm mục tiêu cung cấp một lớp vật lý (PHY) và lớp điều khiển truy cập môi trường (MAC) tối ưu cho các ứng dụng truyền thông không dây tầm ngắn, tốc độ thấp và tiêu thụ năng lượng cực thấp. Đây là một trong những tiêu chuẩn nền tảng quan trọng cho các giao thức mạng tầng trên như Zigbee, 6LoWPAN, WirelessHART, Thread, MiWi và các giao thức truyền thông không dây khác được ứng dụng rộng rãi trong lĩnh vực Internet of Things (IoT), nhà thông minh, công nghiệp thông minh, nông nghiệp chính xác, và các hệ thống mạng cảm biến không dây.

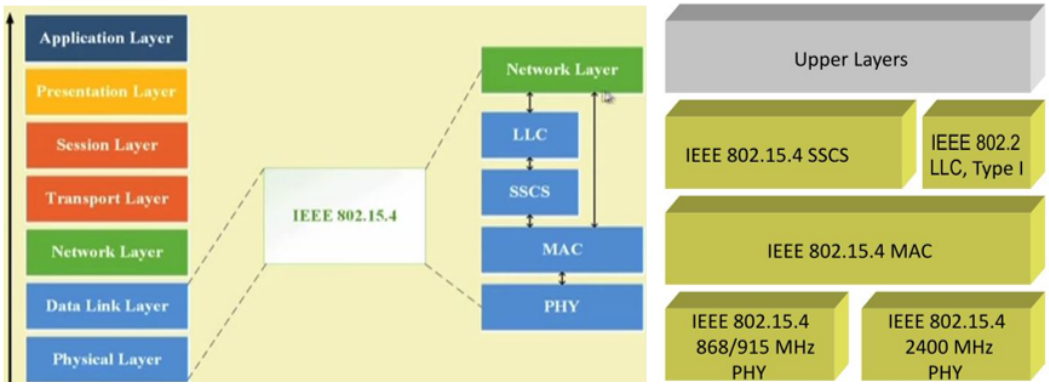
IEEE 802.15.4 được công bố lần đầu tiên vào năm 2003 và đã trải qua nhiều lần cập nhật, hoàn thiện để thích ứng với những thay đổi nhanh chóng trong công nghệ không dây. Mỗi phiên bản cập nhật đều bổ sung các tính năng mới nhằm cải thiện hiệu suất truyền thông, mở rộng dải tần hoạt động, tăng độ tin cậy, mở rộng phạm vi ứng dụng và tăng cường khả năng bảo mật. Tiêu chuẩn này tập trung chủ yếu vào hai lớp trong mô hình OSI: lớp vật lý (Physical Layer – PHY) và lớp liên kết dữ liệu (Data Link Layer – cụ thể là MAC sublayer), nhằm đơn giản hóa thiết kế phần cứng và phần mềm của các thiết bị đầu cuối.

Một trong những điểm mạnh nổi bật của IEEE 802.15.4 là khả năng hỗ trợ tiêu thụ năng lượng thấp. Thiết kế này đặc biệt phù hợp với các thiết bị sử dụng nguồn năng lượng hạn chế như pin hoặc năng lượng thu được từ môi trường (energy harvesting). Các cơ chế như chế độ ngủ sâu (sleep mode), truyền dữ liệu theo khung thời gian (slotted transmission), và cơ chế điều phối mạng (coordinator-based communication) cho phép giảm thiểu đáng kể năng lượng tiêu thụ mà vẫn duy trì độ tin cậy và khả năng truyền nhận dữ liệu ổn định trong môi trường không dây phức tạp.

Về mặt kỹ thuật, IEEE 802.15.4 hỗ trợ nhiều dải tần số khác nhau trên toàn cầu bao gồm 868 MHz (châu Âu), 915 MHz (Bắc Mỹ), và đặc biệt là băng tần ISM 2.4 GHz được sử dụng rộng rãi trên toàn thế giới. Tùy theo dải tần, tốc độ truyền dữ liệu có thể dao động từ 20 kbps (868 MHz) đến 250 kbps (2.4 GHz). Dù tốc độ thấp hơn nhiều so với Wi-Fi hay Bluetooth, nhưng chuẩn này vẫn rất phù hợp cho các ứng dụng truyền dữ liệu không yêu cầu băng thông lớn mà yêu cầu độ tin cậy và ổn định cao, chẳng hạn như giám sát môi trường, thu thập dữ liệu cảm biến, truyền trạng thái thiết bị,. . .

2.1.2 *Mô hình giao thức và tầng được định nghĩa*

Chuẩn IEEE 802.15.4 được thiết kế theo mô hình phân tầng, tuân theo nguyên lý của mô hình OSI (Open Systems Interconnection), tuy nhiên chỉ tập trung định nghĩa và tiêu chuẩn hóa hai tầng dưới cùng là tầng vật lý (Physical Layer - PHY) và tầng điều khiển truy cập môi trường (Medium Access Control - MAC). Đây là hai tầng then chốt chịu trách nhiệm về truyền dẫn dữ liệu không dây giữa các thiết bị trong mạng cảm biến và mạng IoT công suất thấp. Các tầng cao hơn như mạng (Network Layer), vận chuyển (Transport Layer) hay ứng dụng (Application Layer) sẽ do các giao thức tầng trên như Zigbee, Thread hoặc 6LoWPAN định nghĩa và đảm nhận. Cấu trúc phân tầng rõ ràng này giúp cho IEEE 802.15.4 trở thành một nền tảng linh hoạt để phát triển các giao thức truyền thông không dây theo yêu cầu riêng biệt của từng ứng dụng.



Hình 2.1. Cấu trúc phân lớp của IEEE 802.15.4 trong mô hình OSI

2.1.2.1 *Tầng vật lý*

Tầng vật lý trong IEEE 802.15.4 có nhiệm vụ đảm bảo việc truyền nhận bit dữ liệu thô qua môi trường truyền không dây. Nó chịu trách nhiệm mã hóa, giải mã tín hiệu, điều chế, phát và thu sóng vô tuyến, xác định tốc độ dữ liệu, dải tần hoạt động, độ rộng băng thông và công suất phát. Tầng này hỗ trợ ba dải tần chính: 868 MHz, 915MHz và 2.4 GHz.

Bảng 2.1. Các thông số kỹ thuật theo dải tần trong IEEE 802.15.4

Băng tần	Khu vực sử dụng	Kênh	Tốc độ dữ liệu	Điều chế
868 MHz	Châu Âu	1	20 kbps	BPSK
915 MHz	Bắc Mỹ	10	40 kbps	BPSK
2.4 GHz	Toàn cầu	16	250 kbps	O-QPSK + DSSS

Tầng vật lý (PHY) trong chuẩn IEEE 802.15.4 không chỉ đơn thuần đảm nhận nhiệm vụ truyền và nhận bit dữ liệu qua môi trường không dây mà còn cung cấp một loạt các chức năng và dịch vụ hỗ trợ thiết yếu nhằm đảm bảo tính tin cậy, hiệu quả và khả năng thích ứng linh hoạt của hệ thống mạng không dây. Những dịch vụ này

đặc biệt quan trọng trong bối cảnh các hệ thống mạng cảm biến không dây (WSN) hoặc mạng IoT hiện đại yêu cầu tính năng tự tổ chức (self-organizing), tự thích nghi (self-adaptive) và tiêu thụ năng lượng thấp.

Một trong những dịch vụ quan trọng đầu tiên do tầng PHY cung cấp là quét kênh (channel scanning). Trong môi trường mạng không dây, có nhiều kênh truyền khác nhau tương ứng với các tần số riêng biệt trong dải tần cho phép. Thiết bị cần chọn một kênh phù hợp để truyền thông tin – sao cho giảm thiểu xung đột với các thiết bị khác và tránh nhiễu. Quá trình quét kênh cho phép thiết bị rà soát các kênh khả dụng để xác định kênh nào đang rảnh, kênh nào bị chiếm dụng hoặc nhiễu nhiều. Điều này hỗ trợ cho việc thiết lập mạng ban đầu, tái định tuyến khi có thay đổi trong môi trường mạng, hoặc trong các trường hợp mạng tự tổ chức và mở rộng. Quá trình này đặc biệt hữu ích trong các môi trường có mật độ thiết bị cao như thành phố thông minh, nơi nhiều thiết bị cùng hoạt động trong dải tần 2.4 GHz.

Tiếp theo, phát hiện năng lượng tín hiệu (energy detection – ED) là một dịch vụ được tích hợp trong PHY nhằm đánh giá mức năng lượng hiện tại tại một kênh nhất định. Điều này không chỉ hữu ích trong quá trình quét kênh mà còn trong các thuật toán tránh va chạm (collision avoidance) như CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance). Khi thiết bị cần truyền dữ liệu, nó có thể thực hiện phát hiện năng lượng để đánh giá xem kênh có đang được sử dụng hay không. Nếu mức năng lượng cao, có thể kết luận rằng kênh đang bận hoặc có nhiễu, thiết bị sẽ trì hoãn việc truyền để tránh va chạm. Tính năng này góp phần quan trọng trong việc tiết kiệm năng lượng và duy trì hiệu quả truyền dẫn trong mạng.

LQI (Link Quality Indicator) – chỉ số chất lượng liên kết – là một chức năng nổi bật khác của tầng PHY. LQI là một giá trị định lượng, thường được biểu diễn dưới dạng số nguyên trong một dải nhất định (ví dụ 0–255), phản ánh chất lượng tín hiệu của gói tin được nhận. Nó có thể được tính toán dựa trên nhiều yếu tố như tỷ lệ lỗi bit (BER), cường độ tín hiệu nhận (RSSI), hoặc mức độ nhiễu và biến dạng. LQI đặc biệt hữu ích trong các mạng cảm biến dạng lưới (mesh network), nơi thông tin thường được truyền qua nhiều thiết bị trung gian. Với LQI, giao thức tầng trên (như tầng mạng hoặc tầng MAC) có thể lựa chọn tuyến truyền tốt nhất – tức là tuyến có thiết bị trung gian với chất lượng liên kết cao nhất, từ đó tối ưu độ tin cậy và giảm số lần truyền lại do lỗi. Ngoài ra, LQI còn hỗ trợ việc xác định các thiết bị gần kề đáng tin cậy, góp phần vào việc thiết lập bảng định tuyến và phát hiện các nút không ổn định trong mạng.

Bên cạnh các chức năng trên, tầng PHY của IEEE 802.15.4 còn được thiết kế để hoạt động ổn định trong các môi trường có nhiều nhiễu và can thiệp vô tuyến nhờ vào việc áp dụng kỹ thuật điều chế và trải phổ tiên tiến. Cụ thể, đối với băng tần 2.4 GHz – là băng tần phổ biến và được sử dụng trên phạm vi toàn cầu – tầng PHY sử dụng kỹ thuật trải phổ trực tiếp DSSS (Direct Sequence Spread Spectrum) kết hợp với O-QPSK

(Offset Quadrature Phase Shift Keying).

DSSS là kỹ thuật mã hóa tín hiệu trước khi truyền bằng cách nhân tín hiệu dữ liệu với một chuỗi bit giả ngẫu nhiên có tốc độ cao gọi là "chip". Việc này làm trải phổ tín hiệu gốc ra một dải tần rộng hơn, qua đó giúp chống lại nhiễu hẹp băng (narrowband interference) và tăng khả năng phát hiện tín hiệu trong môi trường nhiều nhiễu. DSSS còn hỗ trợ việc giảm xác suất mất gói do xung đột kênh, một vấn đề phổ biến trong môi trường đông thiết bị không dây như trong các ứng dụng nhà thông minh, nông nghiệp thông minh, hay hệ thống giám sát công nghiệp.

O-QPSK là một dạng điều chế pha, trong đó các tín hiệu I và Q (In-phase và Quadrature-phase) được truyền lệch nhau một nửa chu kỳ chip, từ đó làm giảm biến động biên độ tín hiệu và giảm khả năng gây ra nhiễu phổ rộng. Khi kết hợp với DSSS, O-QPSK giúp tín hiệu truyền ổn định, duy trì độ chính xác và khả năng giải điều chế cao ngay cả khi cường độ tín hiệu yếu hoặc trong môi trường có nhiều phản xạ. Nhờ đó, việc sử dụng O-QPSK + DSSS cho tầng PHY trong băng tần 2.4 GHz mang lại hiệu năng vượt trội, đặc biệt phù hợp với các ứng dụng yêu cầu độ tin cậy cao, tốc độ truyền vừa phải (250 kbps), và khả năng hoạt động ổn định trong môi trường phức tạp.

2.1.2.2 Tầng MAC

Tầng MAC (Medium Access Control) trong giao thức IEEE 802.15.4 chịu trách nhiệm kiểm soát quyền truy cập vào môi trường truyền thông không dây, là cầu nối giữa tầng vật lý (PHY) và tầng mạng. Tầng này giữ vai trò then chốt trong việc đảm bảo truyền dữ liệu hiệu quả và tiết kiệm năng lượng, điều đặc biệt quan trọng đối với các ứng dụng mạng cảm biến không dây (WSN) và Internet of Things (IoT), nơi thiết bị thường hoạt động bằng pin trong thời gian dài.

Một trong những chức năng quan trọng của tầng MAC là kiểm soát truy cập kênh truyền thông. Nó sử dụng kỹ thuật đa truy cập phát hiện sóng mang tránh va chạm (CSMA/CA – Carrier Sense Multiple Access with Collision Avoidance). Trước khi gửi dữ liệu, thiết bị sẽ "nghe" kênh để xác định xem có thiết bị khác đang truyền hay không. Nếu kênh đang rảnh, nó sẽ truyền; nếu kênh đang bận, thiết bị sẽ chờ một khoảng thời gian ngẫu nhiên (backoff) rồi thử lại. Điều này giúp giảm nguy cơ va chạm giữa các thiết bị truyền đồng thời trên cùng một kênh.

Tầng MAC xử lý ba loại khung (frame) chính: Data Frame để truyền dữ liệu, Acknowledgment Frame để xác nhận dữ liệu đã nhận, và MAC Command Frame để điều khiển và quản lý hoạt động mạng. Ngoài ra, nó cũng có khả năng gửi hoặc không gửi khung xác nhận tùy theo cài đặt, giúp cân bằng giữa độ tin cậy và tiêu thụ năng lượng.

Trong chế độ hoạt động có beacon (Beacon-enabled mode), tầng MAC triển khai một cấu trúc gọi là superframe. Một siêu khung bắt đầu bằng một beacon được gửi từ Coordinator nhằm đồng bộ thời gian cho toàn mạng. Siêu khung chia thời gian

truyền thành các phần: Beacon, Contention Access Period (CAP), Contention-Free Period (CFP), và Inactive Period. Trong CAP, các thiết bị sử dụng CSMA/CA để truy cập kênh; trong CFP, các thiết bị có thể được cấp thời gian truy cập riêng (GTS – Guaranteed Time Slot); còn Inactive Period là khoảng thời gian các thiết bị có thể tắt radio để tiết kiệm năng lượng.

Tầng MAC cũng đảm nhiệm việc quản lý kết nối của thiết bị với mạng thông qua quá trình Association và Disassociation. Khi một thiết bị muốn gia nhập mạng, nó gửi một yêu cầu Association tới Coordinator. Nếu được chấp nhận, Coordinator sẽ gửi phản hồi cho phép thiết bị trở thành một phần của mạng. Khi thiết bị muốn rời khỏi mạng, nó gửi thông báo Disassociation để giải phóng tài nguyên và duy trì tính nhất quán trong bảng thành viên của mạng.

Bảo mật là một phần tích hợp trong tầng MAC của IEEE 802.15.4. Giao thức hỗ trợ các cơ chế mã hóa dữ liệu bằng thuật toán AES-128, kiểm tra toàn vẹn dữ liệu bằng mã MIC (Message Integrity Code), và xác thực nguồn gửi để chống giả mạo. Tùy theo cấu hình, các khung MAC có thể bao gồm các trường bảo mật giúp bảo vệ dữ liệu khỏi bị nghe lén, thay đổi hoặc tái sử dụng bất hợp pháp.

Một đặc điểm quan trọng của tầng MAC trong IEEE 802.15.4 là khả năng hỗ trợ hoạt động năng lượng thấp. Bằng cách kết hợp cấu trúc siêu khung, khả năng tắt radio trong thời gian không hoạt động, và quản lý truy cập kênh một cách hiệu quả, tầng MAC giúp các thiết bị hoạt động tiết kiệm năng lượng nhưng vẫn duy trì kết nối mạng ổn định. Điều này làm cho IEEE 802.15.4 trở thành một chuẩn truyền thông lý tưởng cho các thiết bị IoT công suất thấp, như cảm biến môi trường, thiết bị đeo, và mạng nhà thông minh.

2.1.2.3 Cấu trúc khung dữ liệu

Tầng vật lý (PHY) của IEEE 802.15.4 chịu trách nhiệm xử lý tín hiệu ở mức phần cứng, bao gồm điều chế, truyền và nhận dữ liệu qua kênh vô tuyến. Gói dữ liệu do tầng PHY xử lý được gọi là PPDU (PHY Protocol Data Unit). Cấu trúc của PPDU bao gồm bốn thành phần chính: Preamble, Start of Frame Delimiter (SFD), PHY Header (PHR) và PHY Service Data Unit (PSDU), trong đó PSDU chính là khung MAC do tầng trên gửi xuống.

Thành phần đầu tiên là Preamble, có độ dài 32 bit (4 byte), được sử dụng để đồng bộ tần số và thời gian giữa máy phát và máy thu trước khi dữ liệu thực sự được truyền đi. Preamble thường là một chuỗi các bit ’0’, cho phép máy thu nhận biết tín hiệu đến và đồng bộ tốc độ bit.

Tiếp theo là SFD (Start of Frame Delimiter), dài 8 bit. Đây là một chuỗi bit xác định điểm bắt đầu chính xác của khung dữ liệu, giúp máy thu phân biệt rõ ràng đâu là phần header và payload.

Sau đó là PHY Header (PHR), dài 8 bit, chứa thông tin điều khiển liên quan đến

khung dữ liệu, chẳng hạn như độ dài của phần dữ liệu (PSDU), cho phép tầng PHY biết được số byte cần tiếp nhận.

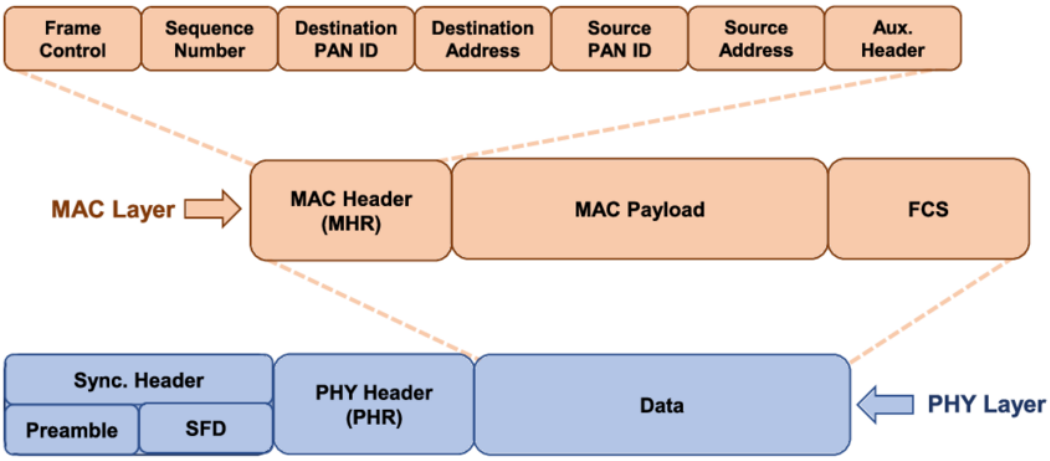
Cuối cùng là PSDU (PHY Service Data Unit), là phần dữ liệu thực tế được truyền đi. Nó bao gồm toàn bộ khung MAC (MAC Header + MAC Payload + FCS) được tạo ra từ tầng MAC. Tầng PHY không xử lý nội dung của PSDU mà chỉ truyền và nhận nó một cách trung lập.Tầng MAC (Medium Access Control) trong IEEE 802.15.4 quản lý việc truy cập kênh truyền, đảm bảo các thiết bị không truyền chồng lấn lên nhau, đồng thời kiểm soát quá trình truyền/nhận gói tin giữa các node trong mạng. Dữ liệu do tầng MAC tạo ra được gọi là MAC Protocol Data Unit (MPDU) và bao gồm ba phần chính: MAC Header (MHR), MAC Payload và Frame Check Sequence (FCS).

MAC Header (MHR) là phần đầu tiên của khung MAC, chứa các trường điều khiển như sau:

- Frame Control (16 bit): xác định loại khung (Beacon, Data, ACK, Command), định dạng địa chỉ (16-bit/64-bit), và các cờ báo hiệu khác như bảo mật, ACK yêu cầu,...
- Sequence Number (8 bit): dùng để đánh số thứ tự khung, giúp nhận dạng và theo dõi các gói dữ liệu (có thể dùng để tránh lặp gói).
- Addressing Fields: bao gồm Destination PAN ID, Destination Address, Source PAN ID và Source Address. Các trường này giúp định tuyến và xác định nguồn/gửi đích của khung. Độ dài các trường này phụ thuộc vào chế độ địa chỉ (short 16-bit hoặc extended 64-bit).
- Auxiliary Security Header (nếu có): chứa các thông tin liên quan đến cơ chế bảo mật như mức độ bảo vệ, chỉ số frame counter,...

Tiếp theo là MAC Payload, chứa dữ liệu thực tế mà ứng dụng hoặc tầng trên gửi xuống. Nội dung của payload thay đổi tùy theo loại khung (Data, Command,...).

Cuối cùng là FCS (Frame Check Sequence), dài 16 bit, là trường kiểm tra lỗi được tính bằng phương pháp CRC-16. Trường này giúp tầng MAC kiểm tra tính toàn vẹn của khung dữ liệu sau khi nhận.



Hình 2.2. Cấu trúc khung dữ liệu tầng PHY và MAC trong IEEE 802.15.4

2.1.3 Các kiến trúc mạng hỗ trợ

Một trong những điểm mạnh quan trọng của chuẩn này là khả năng hỗ trợ nhiều loại kiến trúc mạng khác nhau, giúp đáp ứng các yêu cầu đa dạng của các ứng dụng thực tế. Cụ thể, IEEE 802.15.4 hỗ trợ ba mô hình kiến trúc mạng chính là: mạng sao (Star topology), mạng lưới (Mesh topology) và mạng cây phân cụm (Cluster Tree topology).

2.1.3.1 Mạng sao

- Mạng sao là mô hình đơn giản nhất trong IEEE 802.15.4, nơi toàn bộ mạng được điều phối bởi một thiết bị trung tâm gọi là PAN Coordinator (Personal Area Network Coordinator). Các thiết bị còn lại là thiết bị đầu cuối (End Devices), chúng chỉ có thể gửi và nhận dữ liệu với PAN Coordinator mà không giao tiếp trực tiếp với nhau.
- PAN Coordinator là thiết bị duy nhất có vai trò thiết lập và duy trì mạng, phân phối địa chỉ cho các thiết bị đầu cuối và điều phối truy cập kênh truyền thông trong mạng. Điều này giúp giảm độ phức tạp trong thiết bị đầu cuối, phù hợp với các thiết bị có tài nguyên phần cứng hạn chế và yêu cầu tiêu thụ năng lượng thấp.
- Mạng sao thường sử dụng chế độ Beacon-enabled, trong đó PAN Coordinator phát các beacon định kỳ để đồng bộ thời gian và thông báo trạng thái mạng cho các thiết bị đầu cuối. Điều này rất hiệu quả trong các hệ thống yêu cầu đồng bộ như hệ thống báo động, cảm biến thời gian thực hoặc điều khiển từ xa.
- Mô hình này có ưu điểm là dễ cài đặt, dễ triển khai, chi phí thấp, và tiêu tốn ít năng lượng. Nó phù hợp cho các ứng dụng nhỏ gọn như điều khiển thiết bị gia dụng, mạng cảm biến trong phòng, hệ thống thu thập dữ liệu tập trung ở quy mô hạn chế. Tuy nhiên, nhược điểm của mạng sao là khả năng mở rộng kém và độ tin cậy thấp trong môi trường có nhiễu hoặc khoảng cách truyền xa. Nếu PAN Coordinator bị lỗi hoặc mất kết nối, toàn bộ mạng sẽ ngừng hoạt động. Ngoài

ra, vì các thiết bị đầu cuối không giao tiếp trực tiếp với nhau, nên không thể thực hiện định tuyến hay chia sẻ dữ liệu giữa các nút.

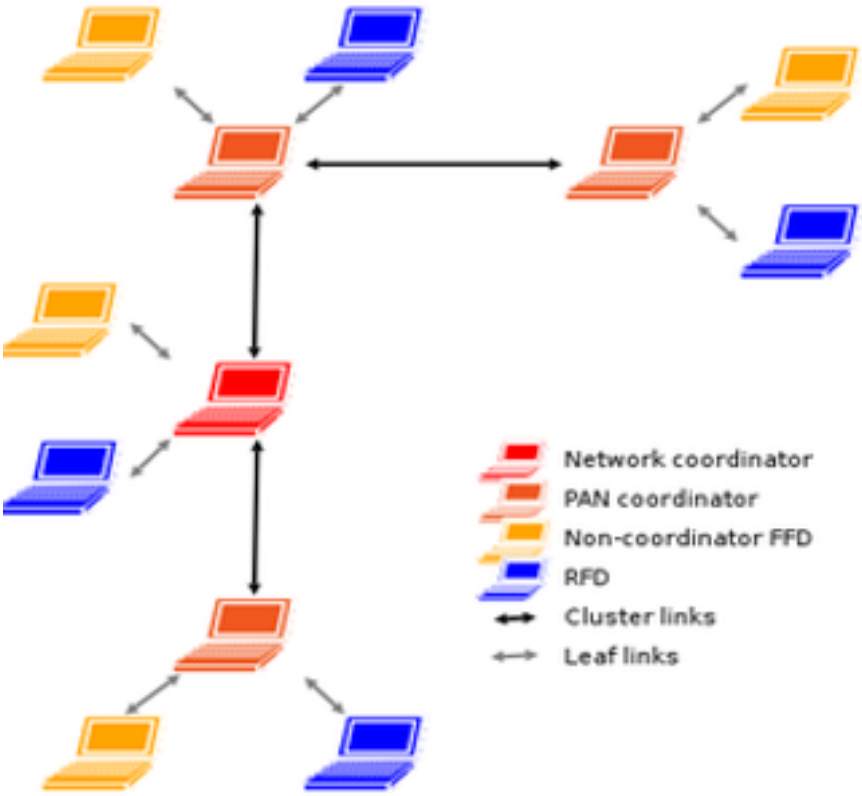
2.1.3.2 Mạng mesh

- Mạng lưới trong IEEE 802.15.4 là mô hình tiên tiến hơn, cho phép các thiết bị trong mạng giao tiếp ngang hàng với nhau và hỗ trợ cơ chế định tuyến nhiều bước (multi-hop routing). Điều này giúp dữ liệu được truyền từ thiết bị này đến thiết bị khác qua các tuyến đường khác nhau để đến đích cuối cùng.
- Mạng bao gồm các thiết bị loại FFD (Full-Function Device) và RFD (Reduced-Function Device). FFD có khả năng tham gia vào quá trình định tuyến và hoạt động như các Router hoặc Coordinator phụ, còn RFD chỉ có thể giao tiếp với một FFD duy nhất và thường đóng vai trò cảm biến hoặc thiết bị đầu cuối với năng lượng thấp.
- Ưu điểm lớn nhất của mạng lưới là khả năng mở rộng cao và độ tin cậy vượt trội. Nếu một đường truyền bị ngắt, mạng có thể tự động tìm tuyến đường thay thế. Điều này đặc biệt quan trọng trong các ứng dụng yêu cầu độ bền mạng cao như giám sát công nghiệp, mạng cảm biến trong nông nghiệp, thành phố thông minh hoặc hạ tầng giao thông.
- Mạng lưới hoạt động hiệu quả trong môi trường động hoặc không ổn định, nơi có thể có nhiều vật cản hoặc nhiễu vô tuyến. Các giao thức cấp cao như Zigbee hoặc Thread thường xây dựng trên kiến trúc này để cung cấp thêm các tính năng định tuyến và bảo mật.
- Nhược điểm của mạng lưới là phức tạp trong thiết kế và triển khai, yêu cầu thiết bị mạnh hơn (FFD), bộ nhớ lớn hơn và phải cài đặt các thuật toán định tuyến như AODV hoặc RPL. Ngoài ra, quản lý đồng bộ và tiết kiệm năng lượng trong mạng lưới là một thách thức lớn do các thiết bị phải duy trì liên lạc liên tục.

2.1.3.3 Mạng cây phân cụm

- Mạng cây phân cụm là một kiến trúc lai giữa mạng sao và mạng lưới, trong đó một PAN Coordinator đóng vai trò gốc của cây, và các Coordinator phụ (cũng là các FFD) mở rộng mạng bằng cách hình thành các cụm (cluster) mới. Mỗi Coordinator phụ có thể điều phối một số thiết bị đầu cuối hoặc các Coordinator phụ khác, tạo thành một cấu trúc cây phân tầng.
- Dữ liệu được truyền trong mạng theo đường dẫn duy nhất từ nút con đến nút cha, và cuối cùng đến PAN Coordinator. Điều này giúp mạng dễ quản lý và có cấu trúc rõ ràng, phù hợp với các ứng dụng phân tầng như giám sát trong tòa nhà, nơi mỗi tầng hoặc khu vực là một cụm riêng biệt.

- Mô hình này hỗ trợ khả năng mở rộng hợp lý bằng cách thêm các cụm con mà không làm gián đoạn hoạt động của mạng tổng thể. Tuy không linh hoạt như mạng lưới, nhưng nó đơn giản hơn trong việc định tuyến và kiểm soát lưu lượng.
- Một ưu điểm nữa là tính ổn định và định hướng rõ ràng trong truyền thông, đặc biệt khi mạng hoạt động trong môi trường ít thay đổi. Việc triển khai theo cụm cũng giúp dễ xác định vị trí lỗi và tối ưu việc phân bổ tài nguyên. Nhược điểm là tính linh hoạt kém: nếu một Coordinator trung gian bị lỗi, tất cả các nút con phía dưới sẽ bị mất kết nối. Ngoài ra, không có khả năng định tuyến lại nên mạng không tự phục hồi tốt như mạng lưới. Do đó, mạng cây phù hợp hơn với các hệ thống cố định và ít thay đổi.



Hình 2.3. Sơ đồ mạng hình cây phân cụm

Bảng 2.2. So sánh các kiến trúc mạng trong IEEE 802.15.4

Tiêu chí	Star	Mesh	Cluster Tree
Số lượng Coordinator	1	Nhiều	1 chính, nhiều phụ
Giao tiếp ngang hàng	Không	Có	Có giới hạn
Định tuyến (Routing)	Không	Có	Có giới hạn
Độ mở rộng	Thấp	Cao	Trung bình
Tính tin cậy	Thấp (1 điểm lỗi)	Cao	Trung bình
Mức độ phức tạp	Thấp	Cao	Trung bình
Ứng dụng tiêu biểu	Nhà thông minh nhỏ	Cảm biến nông nghiệp	Tòa nhà, phân tầng

2.1.4 Các ứng dụng tiêu biểu của IEEE 802.15.4

2.1.4.1 Mạng cảm biến không dây

IEEE 802.15.4 được ứng dụng rộng rãi trong các mạng cảm biến không dây – nơi mà hàng loạt thiết bị cảm biến nhỏ gọn, hoạt động bằng pin được triển khai để thu thập dữ liệu từ môi trường như nhiệt độ, độ ẩm, ánh sáng, áp suất hoặc chuyển động. Với đặc tính tiêu thụ năng lượng thấp và tốc độ truyền dữ liệu vừa đủ, tiêu chuẩn này đặc biệt phù hợp cho các mạng cảm biến cần hoạt động ổn định trong thời gian dài mà không cần thay pin thường xuyên. Mạng WSN sử dụng 802.15.4 thường được triển khai trong nông nghiệp thông minh, giám sát môi trường rừng, đê điều, và theo dõi sức khỏe công trình.

2.1.4.2 Zigbee trong tự động hóa nhà và tòa nhà

Một trong những giao thức nổi bật xây dựng trên nền IEEE 802.15.4 là Zigbee, được sử dụng rộng rãi trong các ứng dụng nhà thông minh và tự động hóa tòa nhà. Với khả năng hình thành mạng mesh, các thiết bị Zigbee như cảm biến chuyển động, đèn, ổ cắm, công tắc,... có thể tự động kết nối và chuyển tiếp dữ liệu lẫn nhau. Điều này giúp người dùng dễ dàng điều khiển thiết bị qua điện thoại hoặc hẹn giờ tự động mà không cần đi dây. Trong các tòa nhà thông minh, Zigbee giúp tiết kiệm năng lượng thông qua điều khiển ánh sáng, điều hòa và giám sát an ninh một cách linh hoạt và hiệu quả.

2.1.4.3 Hệ thống y tế và theo dõi sức khỏe

IEEE 802.15.4 đóng vai trò quan trọng trong các hệ thống theo dõi bệnh nhân không dây, đặc biệt là các mạng Body Area Network (BAN). Các cảm biến đeo trên người bệnh nhân có thể liên tục ghi nhận các thông số y tế như nhịp tim, huyết áp, điện tâm đồ,... và gửi về trung tâm điều khiển để theo dõi từ xa. Với khả năng tiêu

thụ năng lượng cực thấp và hỗ trợ chế độ ngủ sâu, các thiết bị sử dụng 802.15.4 có thể hoạt động lâu dài, giảm số lần sạc pin, từ đó tăng sự tiện lợi và an toàn cho bệnh nhân – đặc biệt là người cao tuổi hoặc bệnh nhân cần giám sát liên tục tại nhà.

2.1.4.4 Hệ thống đo lường thông minh

IEEE 802.15.4 cũng được sử dụng trong các hệ thống đo lường tự động, giúp thu thập dữ liệu từ các đồng hồ điện, nước và gas mà không cần nhân viên đến từng nhà ghi chỉ số. Giao thức Zigbee Smart Energy Profile là một ứng dụng điển hình, cho phép thiết bị đo thông minh giao tiếp hai chiều với trung tâm điều khiển hoặc hệ thống quản lý năng lượng. Từ đó, người dùng có thể nhận được cảnh báo về mức tiêu thụ điện vượt ngưỡng, điều chỉnh hành vi sử dụng năng lượng và thậm chí phối hợp với lưới điện thông minh để tối ưu hóa hiệu suất vận hành.

2.1.4.5 Giao thông thông minh và hạ tầng đô thị số

IEEE 802.15.4 cũng góp phần quan trọng trong các hệ thống giao thông thông minh. Các cảm biến đặt dọc tuyến đường có thể thu thập dữ liệu về lưu lượng xe, tình trạng giao thông, ô nhiễm không khí hoặc hỗ trợ hệ thống đèn tín hiệu thích ứng. Trong các bãi đỗ xe thông minh, cảm biến dựa trên 802.15.4 có thể phát hiện xe vào/ra và báo chỗ trống về trung tâm điều khiển hoặc ứng dụng điện thoại người dùng. Nhờ tiêu chuẩn hỗ trợ mạng mesh và độ tin cậy cao, các hệ thống này hoạt động ổn định trong môi trường đô thị nhiều nhiễu và khoảng cách truyền xa.

2.2 Giao thức truyền thông không dây Zigbee

2.2.1 Tổng quan về Zigbee

Zigbee là một chuẩn truyền thông không dây được thiết kế chủ yếu cho các ứng dụng yêu cầu tiêu thụ năng lượng thấp, tốc độ dữ liệu vừa phải và khả năng kết nối nhiều thiết bị trong một mạng. Chuẩn này được xây dựng dựa trên tầng vật lý và tầng điều khiển truy cập môi trường (MAC) của IEEE 802.15.4, hoạt động trong các băng tần 2.4 GHz, 868 MHz (ở châu Âu) và 915 MHz (ở Bắc Mỹ). Nhờ đặc điểm này, Zigbee đặc biệt phù hợp cho các ứng dụng như nhà thông minh (smart home), hệ thống tự động hóa công nghiệp, giám sát môi trường và các mạng cảm biến không dây.

Một trong những ưu điểm nổi bật của Zigbee là khả năng hỗ trợ mạng mesh (mạng lưới). Với mô hình mạng mesh, các thiết bị trong mạng có thể giao tiếp gián tiếp với nhau thông qua các nút trung gian, giúp mở rộng phạm vi truyền và tăng tính tin cậy. Mỗi thiết bị Zigbee có thể đóng vai trò như một thiết bị điều phối (coordinator), thiết bị router hoặc thiết bị đầu cuối (end device). Kiến trúc này cho phép hàng trăm đến hàng ngàn thiết bị cùng hoạt động trong một mạng Zigbee ổn định, đồng thời tăng khả năng chống mất kết nối khi một thiết bị bị lỗi hoặc hết pin.

Zigbee cung cấp tốc độ truyền dữ liệu khoảng 250 kbps – tuy không cao so với các công nghệ như Wi-Fi hay Bluetooth, nhưng lại là mức lý tưởng cho các ứng dụng cần truyền dữ liệu định kỳ, nhỏ gọn và tiêu thụ điện năng thấp. Một ví dụ điển hình là

các cảm biến đo nhiệt độ, độ ẩm hoặc trạng thái cửa trong nhà thông minh – những thiết bị này thường chỉ cần truyền dữ liệu vài byte mỗi phút hoặc mỗi giờ.

Ngoài ra, Zigbee còn được biết đến với khả năng tiêu thụ năng lượng rất thấp. Các thiết bị đầu cuối có thể được lập trình để "ngủ" phần lớn thời gian và chỉ "thức dậy" khi cần truyền hoặc nhận dữ liệu, từ đó kéo dài thời gian hoạt động lên đến vài năm chỉ với một viên pin nhỏ. Chính nhờ ưu điểm này mà Zigbee trở thành lựa chọn lý tưởng cho các ứng dụng IoT không cần cấp nguồn liên tục. Tính bảo mật trong Zigbee cũng được chú trọng. Chuẩn này sử dụng mã hóa AES 128-bit để bảo vệ dữ liệu truyền tải giữa các thiết bị, đồng thời hỗ trợ các cơ chế xác thực và quản lý khóa nhằm đảm bảo an toàn cho toàn bộ hệ thống.

Bảng 2.3. So sánh một số giao thức truyền thông không dây

Giao thức	Tốc độ	Phạm vi	Năng lượng	Bảo mật	Mở rộng
Zigbee	250 kbps	10–100 m	Rất thấp	AES-128	Rất cao
Wi-Fi	11–1000 Mbps	50–100 m	Cao	WPA2	Trung bình
Bluetooth	1–3 Mbps	10 m	Trung bình	AES-128	Thấp
LoRa	<50 kbps	2–15 km	Rất thấp	AES-128	Cao

Bảng so sánh cho thấy mỗi giao thức không dây có những ưu điểm riêng phù hợp với từng ứng dụng. Zigbee có tốc độ thấp (250 kbps) nhưng tiêu thụ năng lượng rất thấp và khả năng mở rộng mạng cao, thích hợp cho các mạng cảm biến và điều khiển tự động trong nhà. Wi-Fi tuy có tốc độ rất cao (11–1000 Mbps) nhưng tiêu tốn nhiều năng lượng, phù hợp cho các ứng dụng cần băng thông lớn như truyền video.

Bluetooth có tốc độ trung bình (1–3 Mbps), phạm vi ngắn (10 m) và năng lượng tiêu thụ ở mức trung bình, phù hợp với các thiết bị cá nhân như tai nghe hoặc cảm biến đeo. Trong khi đó, LoRa nổi bật với phạm vi truyền xa (2–15 km) và tiêu thụ năng lượng rất thấp, dù tốc độ truyền rất thấp (<50 kbps), nên rất thích hợp cho các ứng dụng giám sát từ xa như nông nghiệp thông minh hoặc thành phố thông minh.

Về bảo mật, cả Zigbee, Bluetooth và LoRa đều sử dụng mã hóa AES-128, đảm bảo an toàn ở mức cao. Wi-Fi dùng chuẩn WPA2, cũng đáp ứng yêu cầu bảo mật tốt trong mạng gia đình và công nghiệp. Tổng thể, Zigbee là lựa chọn cân bằng giữa năng lượng, bảo mật và khả năng mở rộng cho các ứng dụng mạng cảm biến tầm ngắn.

2.2.2 Kiến trúc giao thức Zigbee

Giao thức Zigbee được xây dựng dựa trên mô hình phân lớp tương tự như mô hình OSI (Open Systems Interconnection), trong đó mỗi lớp đảm nhiệm một chức năng riêng biệt, từ truyền tín hiệu vật lý đến xử lý dữ liệu ứng dụng. Mô hình này bao gồm năm lớp chính: lớp vật lý (PHY), lớp điều khiển truy cập môi trường (MAC), lớp mạng (NWK), lớp hỗ trợ ứng dụng (APS) và lớp ứng dụng (APL).

2.2.2.1 Lớp vật lý

Lớp vật lý là tầng thấp nhất trong mô hình Zigbee, đảm nhiệm việc truyền và nhận dữ liệu thô qua môi trường truyền dẫn không dây. Cụ thể, lớp này định nghĩa các đặc tả về tần số hoạt động, sơ đồ điều chế, tốc độ truyền dữ liệu, mức công suất phát, và các đặc tính vật lý khác của tín hiệu. Zigbee sử dụng sơ đồ điều chế O-QPSK (Offset Quadrature Phase-Shift Keying) trong băng tần 2.4 GHz, cho phép truyền dữ liệu ổn định với mức tiêu thụ năng lượng thấp. Ngoài ra, nó còn sử dụng mã hóa dữ liệu và kỹ thuật kiểm soát công suất để tối ưu hóa hiệu năng truyền dẫn.

Zigbee hỗ trợ ba dải tần chính: băng tần 2.4 GHz (toàn cầu) với 16 kênh (CH11 đến CH26) cho tốc độ truyền lên đến 250 kbps; băng tần 915 MHz (chủ yếu tại Bắc Mỹ) với 10 kênh, tốc độ 40 kbps; và băng tần 868 MHz (tại châu Âu) với 1 kênh duy nhất, tốc độ 20 kbps. Khoảng cách truyền dữ liệu trong mạng Zigbee dao động từ vài mét đến hàng trăm mét, tùy thuộc vào công suất phát, môi trường xung quanh (vật cản, nhiễu), và vị trí lắp đặt thiết bị.

2.2.2.2 Lớp MAC

Lớp MAC chịu trách nhiệm điều phối quyền truy cập của các thiết bị vào môi trường truyền dẫn chung, đảm bảo rằng các gói tin không bị va chạm khi nhiều thiết bị cùng gửi dữ liệu. Nó thực hiện việc đóng gói dữ liệu thành các khung (frame), cấp phát địa chỉ MAC, xác thực đơn giản giữa các thiết bị, và hỗ trợ cơ chế tái truyền gói tin khi có lỗi xảy ra trong quá trình truyền.

Lớp MAC trong Zigbee hỗ trợ hai chế độ hoạt động chính: beacon-enabled mode và non-beacon mode. Ở chế độ beacon-enabled, thiết bị điều phối (coordinator) sẽ phát các beacon định kỳ để đồng bộ hóa các thiết bị con, rất phù hợp cho các mạng có cấu trúc phân cấp, tiết kiệm năng lượng. Trong khi đó, non-beacon mode hoạt động dựa trên cơ chế CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance), cho phép các thiết bị truyền dữ liệu khi phát hiện kênh truyền đang rảnh, tránh va chạm. Ngoài ra, lớp MAC còn hỗ trợ chế độ ngủ (sleep mode), cho phép các thiết bị giảm tiêu thụ năng lượng trong những khoảng thời gian không hoạt động.

2.2.2.3 Lớp mạng

Lớp mạng trong Zigbee giữ vai trò quan trọng trong việc hình thành, cấu hình, quản lý và duy trì mạng lưới các thiết bị không dây. Nó phụ trách định tuyến gói tin giữa các thiết bị, cấp phát địa chỉ mạng (network address), và đảm bảo tính ổn định của toàn mạng. Zigbee hỗ trợ các cấu trúc mạng như star (hình sao), tree (cây) và đặc biệt là mesh (mắt lưới), trong đó các thiết bị có thể chuyển tiếp dữ liệu qua nhau để mở rộng phạm vi mạng.

Về định tuyến, lớp mạng hỗ trợ cả hai phương pháp: định tuyến dựa trên bảng (table-based routing), nơi mỗi thiết bị duy trì bảng định tuyến riêng, và định tuyến theo yêu cầu (on-demand routing), nơi đường đi được thiết lập chỉ khi cần thiết. Nhờ khả năng tự cấu hình và tự phục hồi, Zigbee có thể duy trì hoạt động ổn định ngay cả khi

một số thiết bị trong mạng bị lỗi hoặc thay đổi vị trí.

2.2.2.4 Lớp hỗ trợ ứng dụng

Lớp APS đóng vai trò trung gian giữa lớp mạng và lớp ứng dụng, chịu trách nhiệm phân phối dữ liệu giữa các thiết bị ứng dụng và quản lý thông tin liên kết. Nó duy trì bảng liên kết (binding table), trong đó ánh xạ các endpoint (điểm cuối của thiết bị ứng dụng) với các chức năng cụ thể. Nhờ đó, lớp APS có thể điều phối việc truyền dữ liệu một cách đáng tin cậy giữa các thiết bị với nhau.

Lớp này cũng cung cấp các dịch vụ như phân mảnh gói tin nếu dữ liệu vượt quá kích thước tối đa cho phép, hoặc tái kết hợp gói tin khi nhận về. Ngoài ra, lớp APS còn xử lý các yêu cầu dịch vụ từ ứng dụng, hỗ trợ bảo mật dữ liệu trong quá trình truyền và tạo nền tảng để các ứng dụng người dùng hoạt động hiệu quả trên mạng Zigbee.

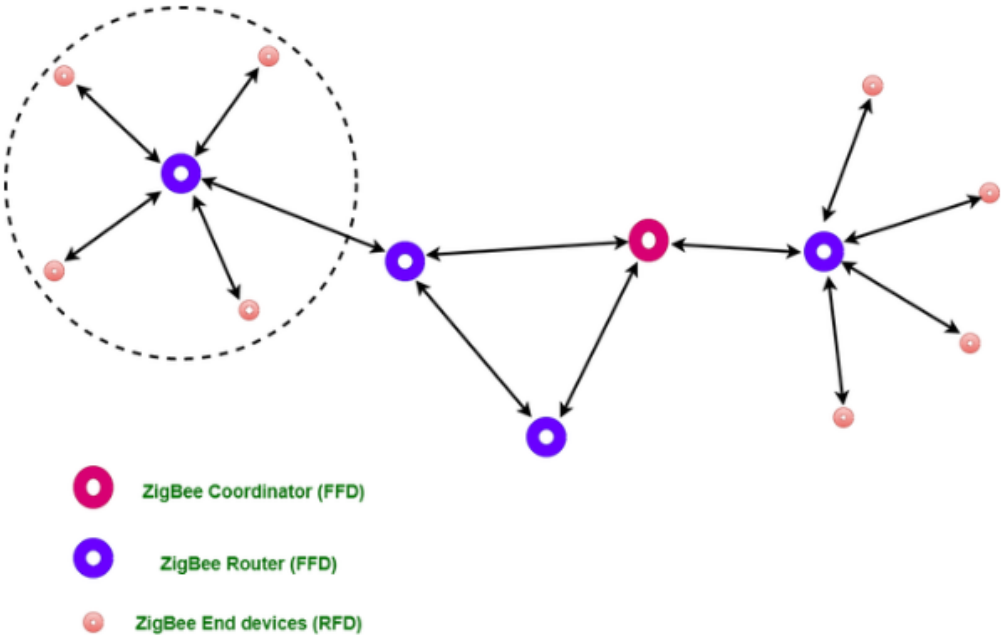
2.2.2.5 Lớp ứng dụng

Lớp ứng dụng là tầng cao nhất trong mô hình Zigbee, nơi triển khai các chức năng cụ thể mà người dùng cuối tương tác. Nó bao gồm hai thành phần chính: Zigbee Device Object (ZDO) và các Application Object (đối tượng ứng dụng). ZDO quản lý các nhiệm vụ thiết yếu như khởi tạo thiết bị, khám phá và kết nối dịch vụ, xác định vai trò của thiết bị (coordinator, router, end device) trong mạng Zigbee.

Mỗi thiết bị Zigbee có thể chứa nhiều Application Object, tương ứng với các chức năng cụ thể như cảm biến nhiệt độ, công tắc, bộ điều khiển đèn, v.v. Các chức năng này được mô tả thông qua các Cluster ID – tập hợp các lệnh và thuộc tính dùng chung cho một loại dịch vụ nhất định. Lớp ứng dụng cho phép thiết bị giao tiếp theo chuẩn Zigbee Application Framework, đảm bảo tính tương thích và khả năng mở rộng của hệ thống.

2.2.2.6 Các loại thiết bị trong mạng Zigbee

Trong mạng Zigbee, các thiết bị được phân loại dựa trên vai trò và chức năng của chúng trong mạng. Có ba loại thiết bị chính trong một mạng Zigbee: Thiết bị điều phối (Zigbee Coordinator - ZC), Thiết bị định tuyến (Zigbee Router - ZR) và Thiết bị đầu cuối (Zigbee End Device - ZED). Mỗi loại thiết bị đảm nhận một vai trò khác nhau trong việc hình thành, duy trì và vận hành mạng. Việc phân chia này cho phép tối ưu hóa tiêu thụ năng lượng, hiệu suất truyền thông và khả năng mở rộng mạng.



Hình 2.4. Mạng Zigbee mesh với Coordinator, Router và End Device

- Zigbee Coordinator (ZC) – Thiết bị điều phối:

  Zigbee Coordinator là thiết bị quan trọng nhất trong mạng Zigbee. Mỗi mạng chỉ có duy nhất một thiết bị điều phối, chịu trách nhiệm khởi tạo mạng, lựa chọn kênh truyền thông, gán địa chỉ mạng, và quản lý thông tin an ninh như khóa mã hóa mạng. ZC cũng có khả năng lưu trữ bảng định tuyến và thông tin thiết bị trong toàn mạng.

  Ngoài chức năng điều phối, ZC có thể thực hiện nhiệm vụ truyền nhận dữ liệu giống như các thiết bị khác. Trong một số ứng dụng như nhà thông minh, ZC thường được kết nối với một máy chủ trung tâm hoặc gateway để thu thập và điều khiển toàn bộ hệ thống. Vì đảm nhiệm vai trò trung tâm và cần hoạt động liên tục, ZC thường được cấp nguồn không giới hạn (kết nối điện lưới thay vì pin).

- Zigbee Router (ZR) – Thiết bị định tuyến:

  Zigbee Router là những thiết bị trung gian có khả năng truyền tiếp gói tin, mở rộng phạm vi mạng và định tuyến dữ liệu giữa các nút khác nhau. Không giống như thiết bị đầu cuối, ZR có khả năng lưu trữ bảng định tuyến và xử lý các yêu cầu định tuyến động. Các router đóng vai trò duy trì kết nối giữa các phần khác nhau trong mạng, đặc biệt quan trọng trong mạng có cấu trúc mesh hoặc tree, nơi gói tin cần truyền qua nhiều nút trung gian để đến đích.

  ZR cũng có thể đóng vai trò như một thiết bị đầu cuối để thu thập hoặc điều khiển dữ liệu. Trong thực tế, các thiết bị như công tắc, bộ điều khiển trung gian hoặc các điểm nút cảm biến cố định có thể hoạt động như một ZR. Vì có trách nhiệm

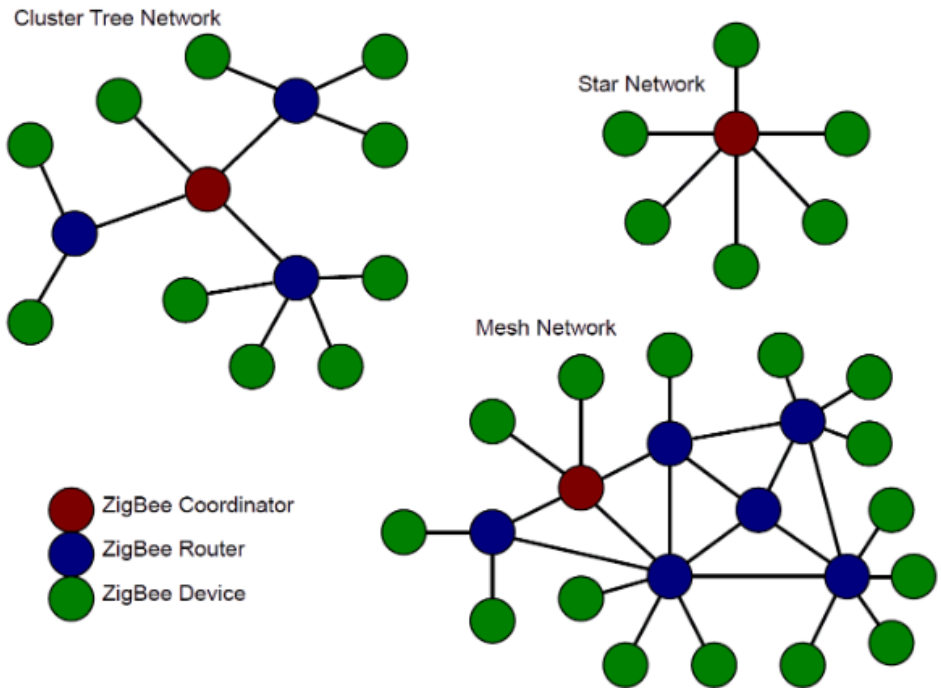
định tuyến, các router cần hoạt động liên tục và cũng thường được cấp nguồn ổn định.

- Zigbee End Device (ZED) – Thiết bị đầu cuối:

  Zigbee End Device là thiết bị đơn giản và tiết kiệm năng lượng nhất trong mạng Zigbee. ZED không thực hiện chức năng định tuyến và chỉ giao tiếp với một thiết bị cha (thường là ZR hoặc ZC). Điều này giúp giảm yêu cầu về bộ nhớ và xử lý, đồng thời tiết kiệm pin cho ZED – một yếu tố rất quan trọng trong các ứng dụng như cảm biến nhiệt độ, cảm biến chuyển động, hoặc công tắc pin.

  ZED có thể được cấu hình để hoạt động ở chế độ ngủ (sleep mode) phần lớn thời gian và chỉ thức dậy định kỳ để gửi dữ liệu hoặc nhận lệnh. Do đó, thời gian sử dụng pin của ZED có thể kéo dài từ vài tháng đến nhiều năm, tùy vào tần suất hoạt động và loại cảm biến.

2.2.2.7 Các cấu trúc mạng Zigbee



Hình 2.5. Các loại kiến trúc mạng Zigbee

- Mạng hình sao

  Trong kiến trúc hình sao, tất cả các thiết bị đầu cuối (Zigbee End Devices - ZED) đều kết nối trực tiếp với một thiết bị điều phối trung tâm gọi là Zigbee Coordinator (ZC). Trong mô hình này, ZC đóng vai trò như một trung tâm điều khiển, chịu trách nhiệm giao tiếp và quản lý toàn bộ các thiết bị con. Do thiết kế đơn giản, mô hình này dễ triển khai và cấu hình, đồng thời tiêu tốn ít tài nguyên định tuyến. Tuy nhiên, điểm yếu lớn nhất của kiến trúc hình sao là sự phụ thuộc hoàn toàn

vào ZC: nếu thiết bị trung tâm bị lỗi, toàn bộ mạng sẽ ngừng hoạt động. Kiến trúc này phù hợp cho các ứng dụng quy mô nhỏ như mạng cảm biến trong nhà, điều khiển đèn chiếu sáng hay các thiết bị gia dụng thông minh.

- Mạng hình cây

  Kiến trúc cây lại sử dụng một cấu trúc phân cấp, trong đó ZC là nút gốc, các Zigbee Router (ZR) hoạt động như các nút trung gian, và các ZED nằm ở lá cây. Mỗi router có thể quản lý một số lượng nhất định các nút con, giúp mạng dễ dàng được mở rộng theo nhánh. Dữ liệu sẽ truyền theo hướng đi rõ ràng dọc theo các nhánh của cây. Ưu điểm của mô hình này là dễ quản lý, cho phép mở rộng phạm vi mạng mà không cần tăng công suất truyền. Tuy nhiên, kiến trúc cây dễ bị gián đoạn nếu một router trung gian gặp lỗi vì các thiết bị con sẽ mất kết nối. Đây là mô hình phù hợp cho các ứng dụng giám sát môi trường, chiếu sáng ngoài trời hoặc mạng cảm biến trong các khu công nghiệp có cấu trúc định sẵn.

- Mạng mesh

  Kiến trúc mesh là mô hình linh hoạt và mạnh mẽ nhất trong Zigbee. Trong mạng mesh, các Zigbee Router có thể giao tiếp trực tiếp với nhiều router khác, tạo thành một mạng lưới liên kết phức tạp. Dữ liệu có thể truyền qua nhiều đường đi khác nhau giữa các nút, giúp tăng tính chịu lỗi và khả năng tự phục hồi của mạng. Nhờ sử dụng các thuật toán định tuyến động như AODV, mạng mesh có thể tự động tìm đường đi tối ưu khi có nút gặp sự cố hoặc khi mạng thay đổi cấu trúc. Mặc dù phức tạp hơn và tiêu tốn năng lượng nhiều hơn so với các kiến trúc khác, mesh là lựa chọn tối ưu cho các ứng dụng quy mô lớn, yêu cầu độ tin cậy cao như nhà thông minh toàn diện, hệ thống an ninh hoặc mạng cảm biến trong công nghiệp.

2.2.3 Định tuyến trong mạng Zigbee

Định tuyến là một thành phần quan trọng trong mạng Zigbee, đặc biệt trong các kiến trúc tree và mesh nơi mà các gói dữ liệu cần phải đi qua nhiều thiết bị trung gian để đến được đích. Zigbee sử dụng các cơ chế định tuyến được tiêu chuẩn hóa trong lớp mạng (Network Layer), cho phép thiết bị tự động tìm đường đi tối ưu và duy trì kết nối ổn định trong môi trường mạng không dây, nơi các nút có thể di chuyển, rơi rụng hoặc thay đổi trạng thái.

2.2.3.1 Bảng định tuyến và thông tin mạng

Mỗi thiết bị Zigbee có khả năng định tuyến (tức Zigbee Coordinator hoặc Zigbee Router) đều duy trì một bảng định tuyến (routing table) và các thông tin cấu trúc mạng như địa chỉ cha, con, bảng láng giềng (neighbor table), bảng chuyển tiếp (route discovery table),... Những bảng này được sử dụng để xác định đường đi tốt nhất của gói tin đến đích. Việc cập nhật bảng định tuyến diễn ra tự động thông qua các quá trình phát hiện nút lân cận và trao đổi thông tin mạng định kỳ.

2.2.3.2 Các thuật toán định tuyến phổ biến

- Định tuyến AODV (Ad-hoc On-demand Distance Vector):

  AODV là thuật toán định tuyến theo yêu cầu, nghĩa là tuyến đường chỉ được thiết lập khi có nhu cầu truyền dữ liệu. Khi một thiết bị cần gửi gói tin đến một thiết bị khác mà không có tuyến sẵn, nó sẽ phát ra một Route Request (RREQ). Các thiết bị trung gian sẽ chuyển tiếp gói này và nếu đích nhận được, sẽ trả lời bằng Route Reply (RREP). Sau đó, tuyến đường được thiết lập và lưu vào bảng định tuyến.

  Ưu điểm của AODV là tiết kiệm tài nguyên vì không cần duy trì liên tục bảng định tuyến đầy đủ cho toàn mạng. Nhược điểm là có độ trễ khi lần đầu thiết lập đường truyền. AODV thường được sử dụng trong mạng mesh Zigbee nhờ khả năng tự phục hồi khi thay đổi cấu trúc mạng.

- Định tuyến theo cây (Tree Routing):

  Mô hình định tuyến cây sử dụng cấu trúc phân cấp, trong đó mỗi nút biết được nút cha của mình và các nút con. Dữ liệu được truyền theo nhánh từ nút con đến nút cha cho đến khi đến được đích hoặc đạt đến một nút có thể định tuyến đến đích.

  Ưu điểm của tree routing là đơn giản, không yêu cầu lưu trữ bảng định tuyến phức tạp. Tuy nhiên, nhược điểm là không linh hoạt: nếu một nhánh của cây bị hỏng, các thiết bị phụ thuộc vào nhánh đó có thể mất kết nối. Tree routing phù hợp cho các ứng dụng có cấu trúc mạng ổn định, ít thay đổi.

- Định tuyến mesh (Mesh Routing):

  Mesh routing là mô hình định tuyến động và linh hoạt nhất. Mỗi thiết bị router có thể kết nối với nhiều thiết bị lân cận và chuyển tiếp gói tin theo các đường đi khác nhau. Kết hợp với AODV, mạng có thể tự động cập nhật tuyến đường khi có sự thay đổi trong cấu trúc mạng hoặc khi có nút gặp lỗi.

  Ưu điểm chính của mesh routing là độ tin cậy cao, khả năng phục hồi mạnh và dễ mở rộng. Dù tiêu tốn tài nguyên hơn, đây là lựa chọn lý tưởng cho các ứng dụng công nghiệp hoặc nhà thông minh với nhiều thiết bị và yêu cầu tính liên tục cao.

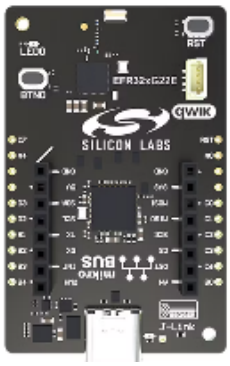
2.2.3.3 Cơ chế xử lý lỗi và phục hồi

Khi một nút trong tuyến đường bị mất hoặc không phản hồi, lớp mạng Zigbee sẽ khởi tạo lại quá trình khám phá đường đi mới bằng AODV hoặc gửi thông báo lỗi để các nút liên quan cập nhật lại bảng định tuyến. Nhờ vậy, mạng có khả năng tự phục hồi, đảm bảo tính ổn định và độ tin cậy trong truyền thông.

2.3 SoC EFR32MG24

2.3.1 Giới thiệu chung về EFR32MG24

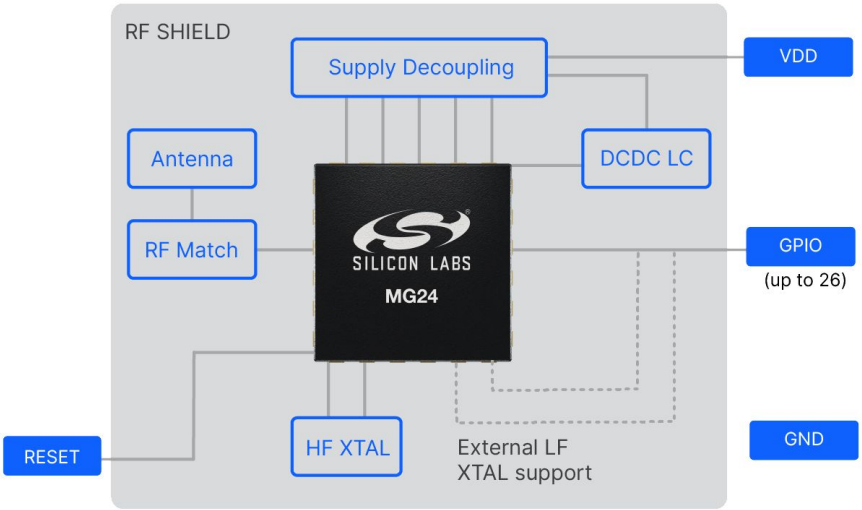
EFR32 là dòng vi điều khiển tích hợp RF (SoC – System on Chip) do hãng Silicon Labs phát triển, được thiết kế chuyên biệt cho các ứng dụng kết nối không dây công suất thấp như Zigbee, Thread, Bluetooth Low Energy (BLE), và chuẩn IEEE 802.15.4. Dòng EFR32 nổi bật nhờ khả năng tích hợp cao, tiêu thụ năng lượng thấp, hiệu suất truyền nhận mạnh mẽ, và bộ công cụ phần mềm mạnh mẽ đi kèm như Simplicity Studio, RAIL, Gecko SDK.



Hình 2.6. Soc EFR32MG24

EFR32MG là dòng chip hỗ trợ đa giao thức không dây, trong đó "MG" viết tắt của "Multiprotocol, General purpose", cho phép truyền thông qua nhiều chuẩn khác nhau, đặc biệt là Zigbee và Thread. Trong số đó, EFR32MG24 là một trong những dòng mới nhất, được tối ưu hóa cho mạng IoT công suất thấp và bảo mật cao.

2.3.2 Kiến trúc phần cứng EFR32MG24



Hình 2.7. Cấu trúc khối phần cứng module xGM240P sử dụng chip EFR32MG24

EFR32MG24 là một vi mạch SoC (System-on-Chip) do Silicon Labs phát triển, tích hợp khả năng xử lý mạnh mẽ, thu phát sóng không dây và nhiều ngoại vi hỗ trợ

trong một chip duy nhất. Trái tim của EFR32MG24 là vi xử lý ARM Cortex-M33, hoạt động ở tần số lên tới 78 MHz. Kiến trúc Cortex-M33 không chỉ cung cấp hiệu năng cao mà còn hỗ trợ các tính năng bảo mật phần cứng như TrustZone và đơn vị xử lý dấu chấm động (FPU), cho phép xử lý tín hiệu nhanh và an toàn trong các hệ thống nhúng hiện đại.

Về bộ nhớ, EFR32MG24 được trang bị lên đến 1024 KB bộ nhớ Flash và 128 KB RAM, đảm bảo không gian đủ lớn cho các ứng dụng IoT phức tạp và khả năng cập nhật chương trình qua mạng (OTA). Bên cạnh đó, chip còn tích hợp đầy đủ các khối ngoại vi như UART, I2C, SPI, ADC, DAC, bộ định thời và GPIO có thể cấu hình linh hoạt, đáp ứng đa dạng nhu cầu kết nối với cảm biến và thiết bị ngoại vi.

Điểm nổi bật nhất của EFR32MG24 là khối radio tích hợp, hỗ trợ truyền thông không dây trong băng tần 2.4 GHz theo nhiều chuẩn như Zigbee, Thread, Bluetooth Low Energy và đặc biệt là IEEE 802.15.4. Bộ thu phát này cho phép điều chỉnh công suất phát tối đa đến +20 dBm và đạt độ nhạy thu lên đến -104 dBm, giúp đảm bảo kết nối ổn định và tiết kiệm năng lượng. Ngoài ra, chip cũng hỗ trợ nhiều chế độ tiết kiệm điện năng, từ chế độ hoạt động đầy đủ (EM0) đến chế độ ngủ sâu (EM4), phù hợp với các ứng dụng cảm biến yêu cầu hoạt động lâu dài bằng pin.

2.4 RAIL(Radio Abstraction Interface Layer)

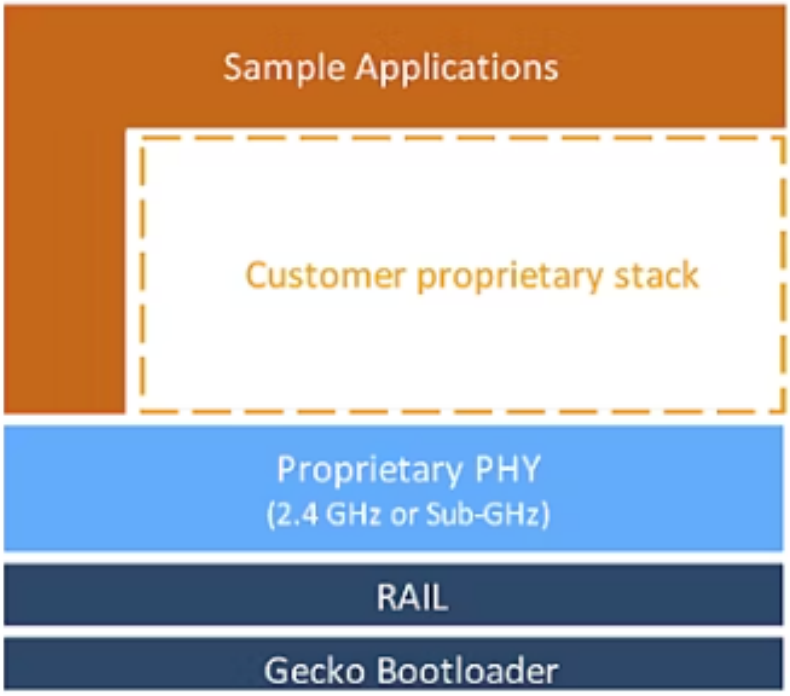
2.4.1 Tổng quan về RAIL

RAIL (Radio Abstraction Interface Layer) là một lớp trừu tượng phần mềm được phát triển bởi Silicon Labs nhằm đơn giản hóa việc điều khiển radio trong các hệ thống nhúng sử dụng dòng SoC EFR32. Thay vì phải thao tác trực tiếp với các thanh ghi phần cứng phức tạp, RAIL cung cấp một giao diện lập trình ứng dụng (API) thống nhất, rõ ràng và dễ sử dụng. Nhờ đó, các nhà phát triển có thể điều khiển toàn bộ hoạt động của radio như truyền, nhận, cấu hình kênh, công suất phát, xử lý sự kiện... chỉ thông qua các lệnh API cấp cao. Những lệnh này sẽ được RAIL chuyển đổi nội bộ thành các thao tác cấp thanh ghi tương ứng để điều khiển phần cứng radio một cách chính xác và hiệu quả.

Một trong những ưu điểm nổi bật của RAIL là khả năng hỗ trợ đa giao thức truyền thông. Nó có thể hoạt động linh hoạt với các giao thức chuẩn như BLE (Bluetooth Low Energy), Zigbee, Thread, Wi-SUN cũng như các giao thức độc quyền (proprietary). Điều này giúp RAIL trở thành nền tảng trung gian lý tưởng để phát triển các ứng dụng không dây tùy chỉnh hoặc dựa trên các chuẩn công nghiệp mà không yêu cầu người lập trình phải hiểu sâu về chi tiết phần cứng cụ thể của từng dòng SoC.

2.4.2 Cấu trúc của RAIL

RAIL được thiết kế với kiến trúc phân tầng, chia thành nhiều lớp riêng biệt nhằm tối ưu hóa khả năng mở rộng, tái sử dụng và dễ bảo trì. Mỗi lớp đảm nhận một chức năng cụ thể trong hệ thống điều khiển radio:



Hình 2.8. Kiến trúc phần mềm RAIL với ngăn xếp tùy chỉnh

- Lớp API (Application Programming Interface):

  Đây là lớp giao diện chính mà ứng dụng người dùng tương tác trực tiếp. Nó cung cấp hàng loạt hàm để điều khiển các chức năng quan trọng của radio như thiết lập kênh truyền, bắt đầu truyền hoặc nhận dữ liệu, xử lý lỗi, cấu hình công suất phát và các tham số vật lý. Tất cả được đóng gói dưới dạng các hàm lập trình rõ ràng và dễ sử dụng.

- Lớp Scheduler (Bộ lập lịch):

  Scheduler trong RAIL đảm nhận vai trò điều phối việc sử dụng tài nguyên radio trong môi trường có nhiều giao thức hoặc nhiều tiến trình truyền thông chạy song song. Nó có khả năng ưu tiên, xếp lịch và chuyển đổi kênh phù hợp để đảm bảo các giao thức không bị xung đột trong quá trình hoạt động, đặc biệt quan trọng trong các ứng dụng đa giao thức.

- Lớp HAL (Hardware Abstraction Layer):

  HAL là lớp trung gian giữa phần mềm và phần cứng. Nó cung cấp các hàm trừu tượng hóa việc truy xuất phần cứng radio, như điều khiển bộ thu phát RF, các bộ định thời, hoặc cấu hình GPIO liên quan đến truyền thông. Nhờ lớp HAL, ứng dụng RAIL có thể chạy trên nhiều dòng SoC khác nhau của Silicon Labs mà không cần thay đổi logic ứng dụng.

- Lớp cấu hình PHY (PHY Configuration):

Lớp này cho phép người phát triển tùy chỉnh các tham số vật lý của radio như tốc độ truyền dữ liệu (data rate), độ rộng băng thông kênh, modulation scheme (ASK, FSK, OQPSK...), công suất phát, độ nhạy thu,... Để hỗ trợ việc cấu hình này, Silicon Labs cung cấp công cụ trực quan gọi là Radio Configurator tích hợp trong Simplicity Studio. Công cụ này cho phép người dùng chọn cấu hình mong muốn mà không cần thao tác tay trên mã nguồn.

2.4.3 Quy trình hoạt động của RAIL

Quy trình hoạt động của RAIL thường bắt đầu bằng bước khởi tạo hệ thống radio, thông qua hàm *RAIL_Init()*. Hàm này sẽ khởi tạo thư viện RAIL, cấp phát các tài nguyên phần cứng cần thiết và đưa hệ thống radio vào trạng thái sẵn sàng. Sau khi khởi tạo, ứng dụng sẽ tiến hành cấu hình radio dựa trên các thông số về PHY như tần số hoạt động, tốc độ truyền, công suất phát và độ rộng băng thông kênh. Việc cấu hình này có thể được thực hiện thủ công thông qua API hoặc thông qua công cụ trực quan Radio Configurator, từ đó sinh ra các tệp cấu hình .c/.h dùng cho chương trình.

Sau khi hoàn tất cấu hình, radio sẽ được đưa vào chế độ nghe (RX) hoặc truyền (TX). Trong chế độ nhận, hàm *RAIL_StartRx()* sẽ được gọi để bắt đầu lắng nghe dữ liệu trên kênh đã chọn. Khi có gói tin đến, RAIL sẽ kiểm tra tính hợp lệ (CRC) và đưa gói tin vào hàng đợi, chờ xử lý. Ngược lại, trong chế độ truyền, ứng dụng sử dụng *RAIL_StartRx()* để phát một gói tin, với dữ liệu được cung cấp từ bộ đệm. Cả hai hoạt động này đều được kiểm soát chặt chẽ bởi RAIL nhằm đảm bảo không có xung đột trong quá trình truyền nhận.

Một thành phần quan trọng trong hoạt động của RAIL là cơ chế xử lý sự kiện (event handling). Khi một sự kiện liên quan đến radio xảy ra, chẳng hạn như gói tin nhận thành công, truyền hoàn tất, kênh bận hoặc lỗi CRC, RAIL sẽ kích hoạt các callback do người dùng đăng ký như *RAILCb_Generic()*để thông báo và xử lý. Điều này giúp ứng dụng chủ động theo dõi và phản ứng với các tình huống trong quá trình truyền thông một cách hiệu quả.

Bên cạnh việc điều khiển truyền nhận, RAIL cũng hỗ trợ quản lý năng lượng tối ưu. Nó cung cấp các API để đưa hệ thống radio về các chế độ tiết kiệm năng lượng như sleep hoặc idle khi không hoạt động, từ đó giúp kéo dài thời lượng pin trong các ứng dụng IoT. Khi cần hoạt động trở lại, RAIL sẽ tự động khôi phục trạng thái radio và tiếp tục truyền nhận như bình thường.

Tổng thể, RAIL cung cấp một quy trình hoạt động chặt chẽ và hiệu quả, bao gồm: khởi tạo $\rightarrow$ cấu hình radio $\rightarrow$ truyền/nhận dữ liệu $\rightarrow$ xử lý sự kiện $\rightarrow$ quản lý năng lượng. Nhờ tính trừu tượng cao, khả năng cấu hình linh hoạt và hỗ trợ đa giao thức, RAIL là một nền tảng lý tưởng cho việc phát triển các ứng dụng không dây trên vi điều khiển của Silicon Labs, đặc biệt trong các hệ thống nhúng cần tối ưu hóa về công suất và hiệu suất truyền thông.

Bảng 2.4. Các API thông dụng trong thư viện RAIL của Silicon Labs

API	**Chức năng**
`RAIL_Init()`	Khởi tạo thư viện RAIL và cấp phát tài nguyên phần cứng. Trả về một `RAIL_Handle_t` dùng cho các API khác.
`RAIL_ConfigChannels()`	Cấu hình các kênh radio dựa trên cấu hình từ Radio Configurator.
`RAIL_ConfigRxOptions()`	Thiết lập các tùy chọn cho chế độ nhận như lọc địa chỉ, tự động xác nhận.
`RAIL_StartRx()`	Bắt đầu chế độ nhận dữ liệu trên kênh chỉ định.
`RAIL_StartTx()`	Bắt đầu truyền dữ liệu với nội dung từ bộ đệm FIFO.
`RAIL_Idle()`	Đưa radio về trạng thái nhàn rỗi (idle).
`RAIL_ConfigEvents()`	Đăng ký các sự kiện radio để nhận qua callback.
`RAIL_ConfigSleep()`	Cấu hình chế độ tiết kiệm năng lượng cho radio.
`RAIL_Sleep()`	Đưa radio vào chế độ sleep.
`RAIL_Wake()`	Đánh thức radio từ chế độ sleep.
`RAIL_GetTxFifoSpaceAvailable()`	Trả về dung lượng trống trong FIFO truyền.
`RAIL_WriteTxFifo()`	Ghi dữ liệu vào FIFO truyền.
`RAIL_ReadRxFifo()`	Đọc dữ liệu từ FIFO nhận.
`RAIL_HoldRxPacket()`	Giữ gói tin đã nhận để xử lý sau.
`RAIL_ReleaseRxPacket()`	Giải phóng gói tin đã giữ sau khi xử lý.
`RAIL_GetRxPacketInfo()`	Lấy thông tin chi tiết về gói tin nhận.
`RAIL_ConfigAutoAck()`	Cấu hình tính năng tự động xác nhận (Auto-ACK).
`RAIL_SetTxPower()`	Thiết lập công suất phát của radio.
`RAIL_GetRssi()`	Đo cường độ tín hiệu nhận được (RSSI).
`RAIL_ConfigData()`	Cấu hình cách quản lý dữ liệu: chế độ PACKET hoặc FIFO.

2.5 Tổng quan về Inverted-F antena và phối hợp trở kháng cho antena

2.5.1 Inverted-F antena

2.5.2 Phối hợp trở kháng cho antena

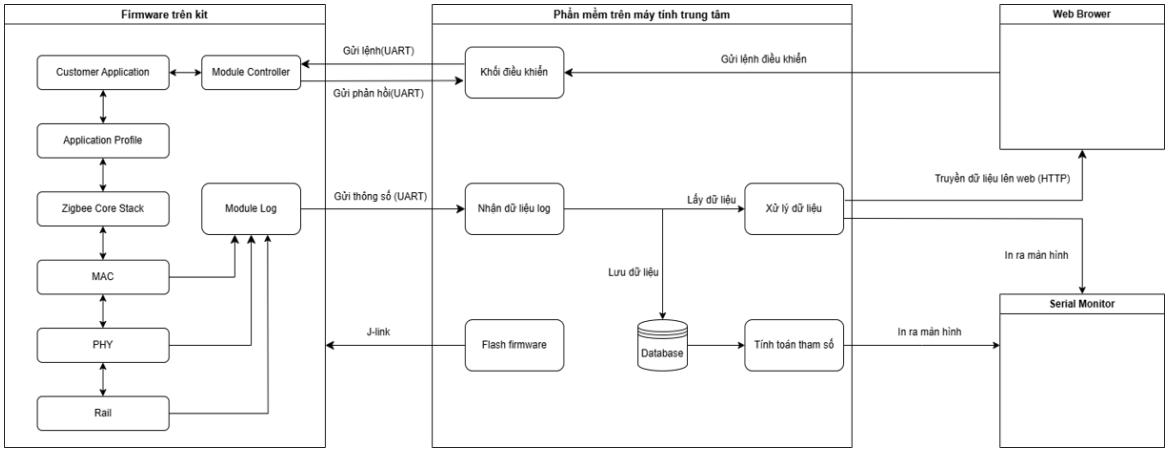
Kết luận chương

CHƯƠNG 3. PHƯƠNG PHÁP LUẬN

Mở đầu chương

3.1 Tổng quan hệ thống

Để xây dựng một môi trường thử nghiệm (Testbed) toàn diện nhằm đánh giá hiệu năng của một hệ thống truyền thông không dây, cần phải thiết kế một kiến trúc bao gồm các khối chức năng cốt lõi, hoạt động hài hòa với nhau.



Hình 3.1. Kiến trúc hệ thống

Hình 3.2 mô tả kiến trúc hệ thống Testbed, được thiết kế theo mô hình Client-Server. Hệ thống được phân chia thành ba khối chức năng chính: Firmware trên kit (đóng vai trò Client, là các thiết bị trong mạng được kiểm thử), Phần mềm trên máy tính trung tâm (đóng vai trò Server, là trạm điều khiển và phân tích) và giao diện người dùng.

3.1.1 Phần mềm nhúng trên kit

Đây là phần mềm nhúng chạy trực tiếp trên các thiết bị phần cứng Zigbee. Trong kiến trúc testbed, khối firmware này trực tiếp chạy các kịch bản kiểm thử trong môi trường vật lý thực tế. Quan trọng hơn, nó là nguồn cung cấp duy nhất những dữ liệu thô và có độ chính xác cao về hoạt động của mạng.

Zigbee Protocol Stack:

- Rail (Radio Abstraction Interface Layer): Đây là lớp trừu tượng hóa phần cứng vô tuyến, nằm ở tầng thấp nhất. Vai trò của Rail là cung cấp một bộ API đồng nhất, cho phép các lớp phần mềm phía trên có thể điều khiển chip vô tuyến mà không cần phải biết về các chi tiết phần cứng phức tạp và đặc thù của nhà sản xuất.
- Lớp PHY và MAC: Lớp PHY chịu trách nhiệm các tác vụ ở mức tín hiệu, như điều chế, giải điều chế và chuyển đổi bit thành sóng vô tuyến để truyền đi. Lớp MAC quản lý việc truy cập vào kênh truyền để tránh xung đột (sử dụng cơ chế

CSMA/CA) và xử lý các gói tin xác nhận (Ack) ở mức liên kết dữ liệu. Các gói tin xác nhận này là cơ sở quan trọng để phát hiện một gói tin có được truyền thành công hay không.

- Zigbee Core Stack: Bao gồm các lớp Mạng (NWK) và Hỗ trợ Ứng dụng (APS). Đây là nơi xử lý các logic phức tạp của mạng Zigbee như hình thành Mạng Khu vực Cá nhân (PAN), định tuyến các gói tin trong mạng mesh, quản lý địa chỉ mạng và các cơ chế bảo mật.

- Application Profile Customer Application: Lớp Application Profile định nghĩa các quy tắc chung cho một loại ứng dụng (ví dụ: chiếu sáng thông minh), còn Customer Application là nơi logic chính của kịch bản thử nghiệm được lập trình viên triển khai. Chính tại đây, các lệnh từ máy tính trung tâm được thực thi, ví dụ như tạo và gửi một số lượng lớn các gói tin dữ liệu theo yêu cầu để kiểm tra hiệu năng mạng.

Các Module chức năng tùy chỉnh:

- Module Controller: Module này liên tục lắng nghe trên cổng giao tiếp nối tiếp UART để nhận các lệnh điều khiển được gửi từ máy tính trung tâm. Sau khi ra lệnh cho Customer Application thực thi, nó sẽ gửi một gói tin phản hồi về máy tính để xác nhận rằng lệnh đã được nhận và xử lý. Cơ chế này tạo ra một vòng lặp điều khiển khép kín, đảm bảo sự đồng bộ và tin cậy giữa giữa các phần trong hệ thống

- Module Log: Đây có thể coi là module quan trọng nhất của toàn bộ hệ thống thu thập dữ liệu. Mỗi khi có một gói tin được truyền đi, nhận về, hoặc bị lỗi, module này sẽ ghi nhận lại các thông số liên quan (ví dụ: trạng thái thành công/thất bại, cường độ tín hiệu nhận được - RSSI, số lần truyền lại). Các thông số này sau đó được đóng gói thành một định dạng nhất quán và gửi liên tục về máy tính trung tâm qua UART, làm đầu vào cho quá trình phân tích hiệu năng.

3.1.2 Phần mềm trên máy tính trung tâm

Đây là một ứng dụng phần mềm có nhiệm vụ điều phối toàn bộ quá trình thử nghiệm, từ việc ra lệnh, thu thập, lưu trữ, xử lý, phân tích cho đến trực quan hóa kết quả cuối cùng.

Khối điều khiển: Là cầu nối nhận lệnh từ người dùng. Khi người dùng thực hiện một thao tác trên giao diện web (ví dụ: nhấn nút "Bắt đầu"), khối này sẽ dịch yêu cầu đó thành một lệnh máy tính cụ thể theo giao thức đã định sẵn và gửi đến Module Controller trên kit qua giao tiếp UART.

Luồng xử lý dữ liệu:

- Nhận dữ liệu log: Một tiến trình chạy nền liên tục lắng nghe trên cổng UART để nhận dòng dữ liệu thông số do Module Log trên kit gửi về. Việc này đảm bảo không một mẩu thông tin nào từ thiết bị bị bỏ lỡ.
- Lưu dữ liệu vào Database: Dữ liệu từ module log sẽ được lưu đầy đủ trong database kèm thời gian tương ứng. Database này lưu giữ tất cả dữ liệu từ lần chạy đầu tiên, người dùng có thể theo dõi, đánh giá và thực hiện tính toán.
- Xử lý dữ liệu: Dữ liệu nhận được qua UART từ module log sẽ được giải mã và lấy ra thông tin từ đó sẽ truyền và hiển thị lên web browser.
- Tính toán tham số: Từ dữ liệu đã được xử lý, khối này áp dụng các công thức và thuật toán thống kê để tính ra các chỉ số hiệu năng quan trọng như PER (Packet Error Rate - Tỷ lệ lỗi gói), PLR (Packet Loss Rate - Tỷ lệ mất gói)

3.1.3 Giao diện và tương tác người dùng

Đây là lớp trên cùng của kiến trúc, là nơi người vận hành và nhà phát triển tương tác với hệ thống, qua 2 cách: giao diện web và serial monitor Web browser:

- Đây là giao diện đồ họa chính, được thiết kế để thân thiện với người dùng. Sử dụng giao diện web là một lựa chọn thiết kế hiện đại, mang lại tính đa nền tảng (có thể truy cập từ bất kỳ hệ điều hành nào) và khả năng điều khiển từ xa qua mạng.
- Thông qua giao diện này, người dùng có thể dễ dàng cấu hình các tham số của kịch bản, ra lệnh điều khiển (bắt đầu, dừng, tạm dừng), và quan trọng nhất là xem kết quả được trực quan hóa dưới dạng các bảng biểu, đồ thị động theo thời gian thực.
- Dữ liệu sau khi được xử lý và tính toán sẽ được truyền từ máy tính trung tâm lên giao diện web thông qua giao thức HTTP.

Serial monitor:

- Đây là một công cụ gỡ lỗi và giám sát cấp thấp, chủ yếu dành cho các lập trình viên và kỹ sư phát triển hệ thống.
- Nó hiển thị trực tiếp dữ liệu thô hoặc các thông báo debug được in ra màn hình từ các khối xử lý. Công cụ này hữu ích để kiểm tra xem dữ liệu có đang được truyền/nhận và xử lý chính xác hay không một cách tức thời, giúp phát hiện và khắc phục sự cố một cách hiệu quả mà không cần đợi dữ liệu hiển thị hoàn chỉnh trên giao diện web.

3.2 Thiết kế phần mềm nhúng trên module EFR32MG24

3.3 Xây dựng phần mềm trên máy tính trung tâm

3.3.1 Xây dựng khối nhận và xử lý dữ liệu qua UART

3.3.1.1 Mục tiêu

Khối nhận và xử lý dữ liệu qua UART là thành phần trung gian giữa phần cứng và phần mềm trung tâm. Nhiệm vụ của khối này là:

- Đọc liên tục dữ liệu gửi về từ KIT qua cổng UART (USB-to-Serial).
- Phân tích, nhận diện và phân loại các loại dữ liệu (gói tin Zigbee, EScan, BER test, trạng thái LED...).
- Giải mã, bóc tách thông tin theo từng lớp giao thức (IEEE 802.15.4, Zigbee Network, Security...).
- Lưu trữ dữ liệu vào cơ sở dữ liệu và cập nhật trạng thái cho các module khác (giao diện, báo cáo, điều khiển).

3.3.1.2 Cách triển khai

Khối UART được triển khai trong lớp Node, chịu trách nhiệm quản lý kết nối với thiết bị phần cứng Zigbee qua cổng COM, khi khởi tạo node, hệ thống sẽ:

- Đóng kết nối cũ nếu đang mở để tránh xung đột.
- Mở kết nối mới với các tham số như cổng COM, tốc độ baud (mặc định 115200), và timeout.
- Tạo một daemon thread riêng để đọc dữ liệu liên tục từ cổng UART, đảm bảo luồng chính không bị block và thread này tự động dừng khi ứng dụng kết thúc.
- Khởi tạo cấu trúc dữ liệu lưu trữ tạm thời như dictionary cho kết quả quét năng lượng (EScan).

Quản lý buffer và phân tách dữ liệu:

- Dữ liệu nhận được từ UART là dạng byte stream liên tục, không có cấu trúc rõ ràng.
- Buffer kiểu bytearray được dùng để lưu trữ tạm thời dữ liệu nhận được.
- Dữ liệu được tách thành từng dòng dựa trên ký tự xuống dòng.
- Mỗi dòng dữ liệu sau khi tách được decode sang chuỗi và chuyển vào pipeline xử lý tiếp theo.

Sau khi tách dòng, hệ thống sử dụng các hàm pattern recognition) để xác định loại dữ liệu nhận được từ UART, giúp phân loại và xử lý đúng chức năng cho từng loại thông tin truyền về từ KIT Zigbee.

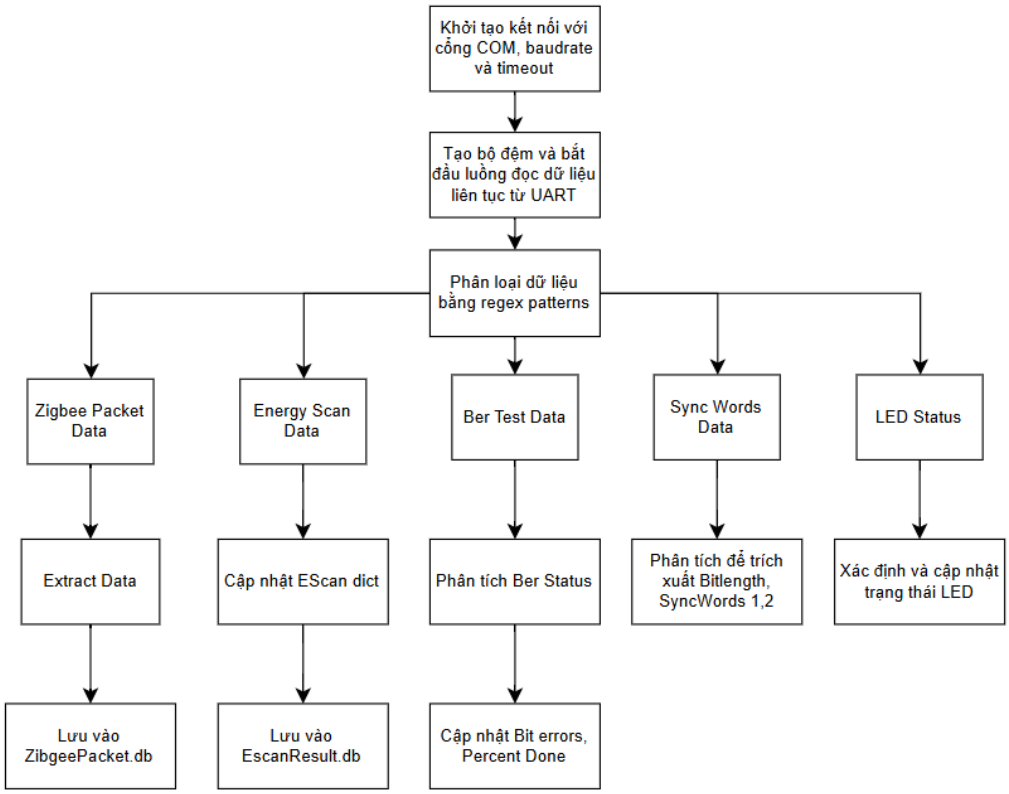
Dữ liệu sự kiện (Event data) là các gói tin Zigbee chứa thông tin về trạng thái mạng, dữ liệu cảm biến hoặc các sự kiện truyền thông, được nhận diện dựa trên từ khóa "Event" xuất hiện trong chuỗi dữ liệu. Khi phát hiện, hệ thống sẽ tạo một đối tượng ZigbeePacket mới, trích xuất các trường thông tin như thời gian, RSSI, LQI, ID64, kênh, payload, sau đó giải mã đa lớp theo chuẩn IEEE 802.15.4 và Zigbee. Quá trình giải mã này bao gồm phân tích Frame Control Field, địa chỉ nguồn và đích, loại khung (Beacon/Data/ACK/Command), sequence number, tiếp tục bóc tách lớp mạng Zigbee (routing, bảo mật, extended addressing) và cuối cùng là lớp payload ứng dụng. Tất cả dữ liệu đã giải mã sẽ được lưu vào cơ sở dữ liệu SQLite với timestamp chuẩn, đồng thời cập nhật vào danh sách packets trong bộ nhớ để phục vụ các module khác.

Dữ liệu quét năng lượng (EScan) là kết quả đo phổ tần trên từng kênh Zigbee, nhận diện qua từ khóa "EScan" và sử dụng regex để trích xuất số kênh và giá trị năng lượng. Sau khi nhận diện, hệ thống xác định ID node (ưu tiên lấy từ packet gần nhất hoặc tạo từ tên cổng), cập nhật giá trị vào dictionary EScan của node và lưu vào bảng escan results trong database. Điều này cho phép người dùng theo dõi chất lượng kênh, phân tích nhiễu và lựa chọn kênh tối ưu cho truyền thông Zigbee.

Dữ liệu kiểm thử BER (Bit Error Rate) là thông tin về tiến trình kiểm thử tỷ lệ lỗi bit giữa hai node, nhận diện qua từ khóa "berstatus". Hệ thống sẽ phân tích các thông số như RSSI, BitErrors, PercentDone, BitsTested, PercentBitError và cập nhật trực tiếp vào thuộc tính của node để phục vụ việc theo dõi tiến trình kiểm thử BER real-time trên dashboard.

Trạng thái đèn LED được nhận diện từ các dòng dữ liệu chứa từ khóa "LedStt" hoặc các thông báo toggle on/off. Hệ thống sẽ cập nhật trạng thái ON/OFF cho node, giúp người dùng kiểm soát và theo dõi trạng thái đèn trên thiết bị từ xa thông qua giao diện web.

Thông tin cấu hình Sync Words là các giá trị đồng bộ tần số radio, nhận diện qua từ khóa "getSyncWords". Sau khi nhận diện, hệ thống sẽ phân tích và cập nhật các giá trị cấu hình này vào node, đảm bảo đồng bộ hóa giữa các thiết bị trong mạng Zigbee.



Hình 3.2. Quy trình xử lý dữ liệu UART

3.3.2 Xây dựng hệ thống cơ sở dữ liệu

3.3.2.1 Giới thiệu và lựa chọn công nghệ

Để đảm bảo hệ thống Testbed có khả năng lưu trữ, truy xuất và phân tích dữ liệu một cách hiệu quả và bền vững, việc xây dựng một hệ thống cơ sở dữ liệu là một yêu cầu bắt buộc. Dữ liệu từ các kịch bản thử nghiệm, bao gồm hàng nghìn gói tin Zigbee và kết quả đo lường, cần được lưu trữ một cách có cấu trúc để phục vụ cho các phân tích sau này cũng như đảm bảo tính toàn vẹn của kết quả nghiên cứu.

Sau khi phân tích các yêu cầu của đồ án, công nghệ SQLite đã được lựa chọn làm hệ quản trị cơ sở dữ liệu chính. Quyết định này dựa trên những ưu điểm vượt trội của SQLite trong bối cảnh một ứng dụng chạy trên máy tính cá nhân như hệ thống Testbed này:

- Kiến trúc không máy chủ (Serverless): Không giống như các hệ CSDL khác như MySQL hay PostgreSQL, SQLite không yêu cầu một tiến trình máy chủ (server process) riêng biệt để hoạt động. Nó được tích hợp trực tiếp vào ứng dụng, giúp hệ thống trở nên gọn nhẹ, giảm thiểu độ phức tạp trong cài đặt và triển khai.
- Lưu trữ trên một tệp tin duy nhất: Toàn bộ CSDL, bao gồm các bảng, chỉ mục và dữ liệu, được lưu trữ trong một tệp tin duy nhất trên đĩa. Điều này làm cho việc sao lưu, di chuyển và quản lý CSDL trở nên vô cùng đơn giản, chỉ cần sao chép một tệp tin là đủ .

- Tích hợp sẵn và dễ sử dụng trong Python: Ngôn ngữ Python đã tích hợp sẵn thư viện sqlite3, cho phép các nhà phát triển tương tác với CSDL SQLite một cách tự nhiên mà không cần cài đặt thêm bất kỳ driver hay thư viện phụ thuộc nào từ bên ngoài .

- Hiệu năng cao cho ứng dụng Desktop: Đối với các ứng dụng có lượng truy cập đồng thời ở mức thấp đến trung bình, SQLite cung cấp hiệu năng đọc và ghi rất cao, hoàn toàn đáp ứng được nhu cầu ghi log gói tin theo thời gian thực của hệ thống.

3.3.2.2 Thiết kế cấu trúc cơ sở dữ liệu

Hệ thống sử dụng 2 file cơ sở dữ liệu sau:

- zigbee_packets.db: Đây được xem là CSDL chính và quan trọng nhất của hệ thống. Nó chịu trách nhiệm lưu trữ thông tin chi tiết của mọi gói tin Zigbee mà hệ thống thu thập được trong quá trình hoạt động. Đây là nguồn dữ liệu gốc để phân tích hiệu năng mạng, gỡ lỗi giao thức và kiểm tra tính đúng đắn của quá trình truyền nhận.

- escan_results.db: Đây là CSDL phụ trợ, được thiết kế chuyên biệt để lưu trữ kết quả từ chức năng quét năng lượng (Energy Scan). Việc tách riêng dữ liệu này giúp cho các truy vấn liên quan đến phân tích phổ tần được thực hiện nhanh chóng và không làm ảnh hưởng đến hiệu suất của CSDL chính.

Bảng 3.1. Cấu trúc bảng zigbee_packets

Tên cột	**Kiểu dữ liệu**	**Mô tả chi tiết và vai trò**
ID	TEXT	Định danh của node đã nhận hoặc gửi gói tin. Trong hệ thống, đây là trường linh hoạt có thể chứa địa chỉ 64-bit (ID64) nếu có.
event	TEXT	Loại sự kiện liên quan đến gói tin, thường là `rxPacket` để chỉ một gói tin đã được nhận thành công.
data	TEXT	Chuỗi dữ liệu thô (payload) của gói tin dưới dạng hệ thập lục phân (hex), được phân tách bằng khoảng trắng. Đây là phần dữ liệu ứng dụng.
time	INTEGER	Dấu thời gian (timestamp) của gói tin do chính firmware trên KIT cung cấp, có độ chính xác cao đến microsecond.
rssi	INTEGER	Cường độ tín hiệu nhận được (Received Signal Strength Indicator), là một chỉ số quan trọng để đánh giá chất lượng tín hiệu, đơn vị dBm.
antena_id	INTEGER	ID của antenna vật lý đã được sử dụng để nhận gói tin này.
sync_word	INTEGER	Giá trị từ đồng bộ (sync word) của radio, hữu ích cho việc gỡ lỗi ở tầng vật lý.
is_ack	INTEGER	Cờ xác định đây có phải là một gói tin báo nhận (ACK) hay không (giá trị 1 hoặc 0).
crc_pass	INTEGER	Cờ xác định gói tin có vượt qua kiểm tra tính toàn vẹn dữ liệu (CRC) hay không (giá trị 1 hoặc 0).
lqi	INTEGER	Chỉ số chất lượng đường truyền (Link Quality Indicator), một thước đo khác về độ tin cậy của liên kết truyền thông.
channel	INTEGER	Kênh tần số mà gói tin được nhận, nằm trong dải tần 2.4 GHz của Zigbee (từ 11 đến 26).
created_at	TIMESTAMP	Thời gian khi bản ghi được thêm vào CSDL.

Bảng 3.1 là nơi lưu trữ chi tiết của mỗi gói tin Zigbee nhận được. Cấu trúc bảng được định nghĩa trong hàm create_database() của mã nguồn.

Bảng 3.2. Cấu trúc bảng escan_data

Tên cột	**Kiểu dữ liệu**	**Mô tả chi tiết và vai trò**
ID64	TEXT	Địa chỉ MAC 64-bit của node đã thực hiện quét năng lượng. Việc sử dụng NOT NULL đảm bảo mỗi kết quả đều được liên kết với một thiết bị cụ thể.
channel	INTEGER	Kênh tần số được quét (từ 11 đến 26).
energy_value	REAL	Giá trị năng lượng đo được trên kênh đó, đơn vị dBm. Kiểu REAL cho phép lưu trữ các giá trị số thực.
scan_timestamp	TIMESTAMP	Thời gian khi kết quả quét được ghi nhận.

Bảng 3.2 được tối ưu cho việc lưu trữ kết quả quét năng lượng, được định nghĩa trong hàm create_escan_database() của mã nguồn.

3.3.2.3 Triển khai và tích hợp vào hệ thống

Các database được khởi tạo ngay sau khi chạy chương trình lần đầu tiên, việc lưu dữ liệu vào CSDL được tích hợp chặt chẽ trong luồng xử lý dữ liệu UART, đảm bảo không có độ trễ và không bỏ sót thông tin.

Đối với gói tin Zigbee:

- Lớp ZigbeePacket có phương thức save_to_db() được gọi ngay sau khi dữ liệu gói tin được trích xuất thành công qua ExtractData().
- Phương thức này thực hiện kết nối tới zigbee_packets.db, chèn dữ liệu gói tin vào bảng zigbee_packets với đầy đủ các trường như ID, event, data, time, rssi, lqi, channel, ...
- Sau khi ghi, kết nối được đóng lại để tránh giữ tài nguyên lâu, đồng thời có xử lý ngoại lệ để đảm bảo hệ thống không bị crash khi gặp lỗi ghi dữ liệu.

Đối với kết quả EScan:

- Hàm save_escan_result() được gọi trực tiếp trong phương thức IsEScan() của lớp Node khi phát hiện dòng dữ liệu quét năng lượng.
- Hàm này trích xuất ID node, kênh, giá trị năng lượng, và lưu vào bảng escan_data trong escan_results.db.
- Việc lưu dữ liệu được thực hiện ngay khi nhận được dữ liệu, giúp dữ liệu quét năng lượng luôn được cập nhật tức thời.

Tất cả các thao tác tương tác với CSDL, đặc biệt là các lệnh INSERT, đều được đặt trong khối try...except để bắt lỗi và xử lý ngoại lệ. Khi có lỗi như tệp CSDL bị khóa, lỗi phân quyền, hay lỗi cú pháp SQL, hệ thống sẽ ghi lại thông báo lỗi chi tiết trên console hoặc log file, giúp việc gỡ lỗi trở nên thuận tiện. Ngoài ra, việc sử dụng with hoặc context manager trong Python giúp tự động đóng kết nối CSDL sau khi thực hiện xong thao tác, tránh rò rỉ tài nguyên.

3.3.3 Xây dựng server Flask

3.3.3.1 Giới thiệu và lựa chọn công nghệ

Để điều khiển, giám sát và tương tác với toàn bộ hệ thống Testbed một cách hiệu quả, việc xây dựng một máy chủ backend (backend server) là yếu tố không thể thiếu. Máy chủ này đóng vai trò là "bộ não" trung tâm, là cầu nối giữa giao diện người dùng (frontend) và lớp logic điều khiển phần cứng. Sau khi phân tích các yêu cầu, công nghệ Flask - một micro-framework mạnh mẽ và linh hoạt của Python - đã được lựa chọn để xây dựng máy chủ này.

Lý do lựa chọn Flask:

- Gọn nhẹ và linh hoạt: Flask cung cấp các công cụ cốt lõi để xây dựng ứng dụng web mà không áp đặt một cấu trúc cứng nhắc, cho phép nhà phát triển toàn quyền quyết định kiến trúc của ứng dụng.
- Dễ học và phát triển nhanh: Cú pháp của Flask đơn giản, dễ tiếp cận, giúp đẩy nhanh quá trình phát triển và gỡ lỗi.
- Hệ sinh thái mạnh mẽ: Flask có một cộng đồng lớn và nhiều tiện ích mở rộng có thể được tích hợp khi cần.
- Tương thích hoàn hảo với Python: Vì toàn bộ lớp logic (Node class, Packet class) đã được viết bằng Python, việc sử dụng Flask tạo ra một hệ thống đồng nhất, dễ dàng tích hợp và chia sẻ dữ liệu giữa các thành phần.

Máy chủ Flask trong đồ án này đảm nhiệm ba vai trò chính:

- Cung cấp các điểm cuối (endpoints) theo chuẩn RESTful để giao diện người dùng có thể gửi lệnh và yêu cầu dữ liệu.
- Quản lý trạng thái của toàn bộ hệ thống, bao gồm danh sách các node đang kết nối, trạng thái của các bài kiểm thử,...
- Điều phối các hoạt động, nhận lệnh từ người dùng và ra lệnh cho các Node object tương ứng để thực thi trên phần cứng.

3.3.3.2 Xây dựng API theo chuẩn RESTful

Nhóm API Quản lý kết nối và node:

- GET /refresh_ports: Cung cấp một phương thức để frontend có thể lấy danh sách các cổng COM đang có sẵn trên máy tính, giúp người dùng dễ dàng lựa chọn cổng để kết nối.
- POST /open_port: Đây là một API có logic xử lý mạnh mẽ. Khi nhận yêu cầu kết nối, nó không chỉ đơn thuần tạo một kết nối mới mà còn kiểm tra xem node với node_id đó đã tồn tại hay chưa. Nếu đã tồn tại, nó sẽ thực hiện quy trình "dọn dẹp" (cleanup): đóng kết nối serial cũ, xóa đối tượng Node cũ ra khỏi bộ nhớ, và chờ một khoảng thời gian ngắn để đảm bảo cổng COM được giải phóng hoàn toàn trước khi tạo kết nối mới. Điều này giúp hệ thống cực kỳ ổn định, tránh các lỗi liên quan đến xung đột tài nguyên.
- POST /close_port và POST /remove_device: Cung cấp các phương thức để người dùng ngắt kết nối và xóa một thiết bị ra khỏi giao diện quản lý một cách an toàn.

Nhóm API điều khiển chức năng Zigbee:

- POST /zigbee_action: Một endpoint đa năng được thiết kế theo mẫu "Command Pattern". Frontend chỉ cần gửi một action (ví dụ: reset, create_network, toggle_led, và API này sẽ gọi đến phương thức tương ứng của đối tượng Node. Thiết kế này giúp mã nguồn phía backend trở nên gọn gàng, dễ bảo trì và mở rộng.
- GET /get_led_state/<node_id>: Một API chuyên biệt để lấy trạng thái đèn LED của một node, phục vụ cho việc cập nhật giao diện theo thời gian thực.

Nhóm API truy xuất và hiển thị dữ liệu:

- GET /get_all_packages/<node_id>: API này được frontend gọi định kỳ (polling) để lấy danh sách tất cả các gói tin đã được thu thập. Nó không chỉ trả về dữ liệu thô mà còn bổ sung thêm các thông tin hữu ích như TypePackage (loại gói tin đã được giải mã) và RealTime (thời gian thực khi API được gọi) để hiển thị.
- POST /get_packet_detail/<node_id>: Một API rất tinh vi, thể hiện độ sâu kỹ thuật của hệ thống. Khi người dùng nhấp vào một gói tin trên giao diện, frontend sẽ gửi thông tin cơ bản của gói tin đó đến API này. Backend sẽ tìm kiếm đối tượng ZigbeePacket đầy đủ trong bộ nhớ và sau đó gọi hàm create_packet_detail() để tạo ra một cấu trúc JSON phân cấp, chi tiết, mô phỏng lại cách mà công cụ phân tích mạng chuyên nghiệp Wireshark hiển thị thông tin. Điều này cho phép người dùng xem chi tiết từng bit của các trường điều khiển, rất hữu ích cho việc gỡ lỗi và nghiên cứu sâu về giao thức.

Nhóm API tích hợp chức năng chuyên biệt (EScan, BER Test):

- EScan: POST /start_escan để bắt đầu quá trình quét và GET /get_escan_data để lấy kết quả và vẽ biểu đồ.
- bertest_open_port, bertest_close_port, bertest_config_sync: Để quản lý kết nối và cấu hình cho các node tham gia bài kiểm thử.
- bertest_start: API quan trọng nhất, nhận các tham số như node phát (TX), node nhận (RX), số lượng bit, tần số. Nó khởi tạo một thread riêng (run_bertest) để chạy tác vụ kiểm thử nặng này trong nền. Đây là một quyết định thiết kế kiến trúc then chốt, giúp giao diện người dùng không bị "đóng băng" trong suốt quá trình kiểm thử có thể kéo dài.
- bertest_status: Một API thông minh để frontend có thể polling. Nó có logic để kiểm tra xem bài test có đang chạy hay không. Nếu đang chạy, nó sẽ lấy dữ liệu live data từ các thuộc tính của node RX. Nếu đã kết thúc, nó sẽ trả về kết quả cuối cùng đã được lưu trữ.
- bertest_stop, bertest_clear_results: Cung cấp các tiện ích để quản lý vòng đời của bài kiểm thử.

3.3.3.3 Quản lý trạng thái và xử lý đồng thời

Quản lý trạng thái toàn cục: Server sử dụng các dictionary toàn cục (nodes, bertest_nodes) và các biến (bertest_running) để lưu trữ trạng thái của toàn bộ hệ thống. Mọi thông tin về các node đang kết nối, trạng thái của chúng, và tiến trình của các bài test đều được lưu trữ ở đây. Đây là một phương pháp quản lý trạng thái đơn giản nhưng hiệu quả cho quy mô của ứng dụng này.

Đảm bảo tính đồng thời: Việc sử dụng threaded=True và các thread riêng cho các tác vụ dài hơi (đọc UART, chạy BER test) là nền tảng cho khả năng xử lý đồng thời của hệ thống. Nó cho phép server có thể vừa chạy một bài test BER, vừa nhận và xử lý các gói tin Zigbee, vừa trả lời các yêu cầu cập nhật giao diện từ nhiều người dùng (nếu có) mà không bị xung đột.

3.4 Xây dựng Engine giải mã các loại gói tin

Trong một hệ thống kiểm thử mạng không dây, việc chỉ thu thập dữ liệu thô là chưa đủ. Để thực sự hiểu được hoạt động của mạng, cần phải có một cơ chế phân tích, từng gói tin được truyền đi. Nó được xây dựng với mục tiêu biến những chuỗi byte hex nhận được từ UART thành thông tin có cấu trúc, phân cấp theo từng lớp giao thức.

3.4.0.1 Tầng IEEE 802.15.4

Beacon Frame được phát ra định kỳ bởi các thiết bị có vai trò Coordinator hoặc Router. Chúng có nhiệm vụ quảng bá sự tồn tại của mạng PAN, cho phép các thiết bị

mới khám phá và xin gia nhập mạng. Engine nhận diện Beacon Frame thông qua 3 bit đầu tiên của Frame Control Field có giá trị 000. Sau khi xác định, quá trình giải mã diễn ra như sau:

- Frame Control Field Analysis: Engine phân tích đầy đủ 16 bit FCF, trích xuất các thông tin như Security Enabled, Frame Pending, ACK Request, PANID Compression, Frame Version, và các chế độ địa chỉ.
- Sequence Number: Đọc số thứ tự của gói tin để theo dõi và phát hiện mất gói.
- Addressing Information: Tùy theo SrcAddrMode, engine sẽ đọc Source PAN ID và địa chỉ nguồn (có thể là địa chỉ ngắn 16-bit hoặc mở rộng 64-bit).
- Payload Processing: Phần payload chứa Superframe Specification và các thông tin cấu hình mạng khác.

Data Frame (Frame Type: 001) là loại gói tin quan trọng và phức tạp nhất, có nhiệm vụ vận chuyển dữ liệu từ tầng ứng dụng qua mạng không dây. Mọi thông tin từ cảm biến, lệnh điều khiển thiết bị, hay dữ liệu trạng thái đều được đóng gói trong Data Frame. Quy trình giải mã đa lớp như sau:

- Tầng IEEE 802.15.4:
  * Frame Control Analysis: Giải mã đầy đủ FCF với 10 trường thông tin chi tiết.
  * Addressing Complete: Xử lý addressing phức tạp bao gồm Destination PAN ID, Destination Address, Source Address với hỗ trợ đầy đủ các chế độ địa chỉ (None, Reserved, Short, Extended).
  * PAN ID Compression Handling: Logic thông minh xử lý việc có/không có Source PAN ID dựa trên cờ PANIDCompression.
- Tầng Zigbee Network: Nếu payload còn đủ dữ liệu, engine sẽ gọi ParseZigbeeNetwork() để bóc tách lớp mạng ZigBee:
  * Network Frame Control: Phân tích 16-bit NWK FCF để xác định FrameType (Data/Command), ProtocolVersion, DiscoverRoute, Multicast, Security flags.
  * Network Addressing: Đọc Destination và Source Network Address (16-bit).
  * Routing Information: Trích xuất Radius (TTL) và Sequence Number của lớp mạng.
  * Extended Features: Xử lý Extended Destination/Source Address (64-bit), Multicast Control, Source Route List nếu có.
- Tầng Zigbee Security: Nếu cờ Security được bật, engine tiếp tục với ParseZigbeeNetworkSecu

    * Security Control Analysis: Phân tích SecurityLevel (0-7), KeyIdentifier, ExtendedNonce flags.
    * Frame Counter Processing: Đọc 4-byte Frame Counter (Little Endian) để chống replay attack.
    * Extended Nonce Handling: Nếu ExtendedNonce=true, đọc thêm 8-byte Source Address.
    * Key Sequence: Xử lý Key Sequence Number nếu KeyIdentifier=1.
    * MIC Processing: Tính toán và xử lý Message Integrity Code dựa trên Security Level: level 0: không mã hóa, không MIC; level 1,5: MIC 32 bit; level 2,6: MIC 64 bit; level 3,7: MIC 128 bit

- Application payload: Sau khi bóc tách tất cả các lớp header, phần dữ liệu còn lại chính là payload ứng dụng (có thể đã được mã hóa).

ACK Frame (Frame Type: 010) là cơ chế nền tảng đảm bảo độ tin cậy trong truyền thông không dây. Khi một thiết bị gửi gói tin có bật cờ "ACK Request", thiết bị nhận sẽ gửi lại ACK để xác nhận đã nhận thành công:

- Chỉ chứa FCF và Sequence Number để tiết kiệm băng thông.
- Không có addressing, ACK được gửi ngay lập tức bởi thiết bị nhận, không cần thông tin địa chỉ phức tạp.
- Xử lý nhanh với việc reset các trường không sử dụng.

Command Frame (Frame Type: 011) được sử dụng cho các tác vụ quản lý và điều khiển ở tầng MAC, không dùng để truyền dữ liệu ứng dụng mà để quản lý kết nối và hoạt động của mạng. Quy trình giải mã:

- Standard Addressing: Xử lý đầy đủ addressing như Data Frame.
- Command ID Recognition: Engine có khả năng nhận diện các lệnh cụ thể:

Bảng 3.3. Các loại Command Frame được hỗ trợ trong hệ thống

Command ID (Hex)	**Tên Lệnh**	**Mô tả Chức năng và Vai trò**
`0x01`	**Association Request**	Yêu cầu gia nhập mạng được gửi từ thiết bị muốn tham gia vào mạng PAN hiện có. Đây là bước đầu tiên trong quá trình thiết lập kết nối giữa thiết bị cuối (End Device/Router) và Coordinator/Router. Gói tin này chứa thông tin về khả năng của thiết bị (device capabilities) và yêu cầu được cấp phát một địa chỉ mạng ngắn (16-bit).
`0x02`	**Association Response**	Phản hồi từ Coordinator/Router đối với Association Request. Gói tin này thông báo kết quả của quá trình gia nhập mạng - thành công hay thất bại. Nếu thành công, nó sẽ chứa địa chỉ mạng ngắn (short address) được cấp phát cho thiết bị mới. Nếu thất bại, nó sẽ chứa mã lỗi (error code) giải thích lý do từ chối.
`0x04`	**Data Request**	Yêu cầu dữ liệu được gửi từ thiết bị con (child device) đến thiết bị cha (parent device - thường là Coordinator hoặc Router). Lệnh này được sử dụng trong cơ chế *indirect transmission*, khi thiết bị cha có dữ liệu chờ gửi cho thiết bị con nhưng thiết bị con đang ở chế độ tiết kiệm năng lượng (sleep mode). Thiết bị con sẽ thức dậy định kỳ và gửi Data Request để kiểm tra có dữ liệu chờ hay không.
`0x07`	**Beacon Request**	Yêu cầu phát Beacon được gửi để khám phá các mạng PAN có sẵn trong vùng lân cận. Khi nhận được lệnh này, tất cả các Coordinator và Router trong tầm nghe sẽ phản hồi bằng cách phát gói tin Beacon chứa thông tin về mạng của họ (PAN ID, địa chỉ, khả năng chấp nhận thiết bị mới, v.v.). Đây là cơ chế quan trọng cho quá trình *network discovery* và *network selection*.

- Intelligent Typing: Engine không chỉ xác định là "Command" mà còn phân loại chính xác loại lệnh cụ thể, ví dụ "Association Request" thay vì "Command(01)".

3.4.0.2 Tầng Zigbee Network

Network Data Frame (NWK Frame Type: 00):

- Routing Management: Xử lý Discover Route với 2-bit value (not just boolean).
- Multicast Support: Phân tích Multicast Control field.
- Source Routing: Giải mã Source Route List với RelayCount và RelayIndex.
- Extended Addressing: Hỗ trợ 64-bit extended addresses cho cả source và destination.

Network Command Frame (NWK Frame Type: 01): Các lệnh quản lý mạng ZigBee như route discovery, leave notification, route record được xử lý với cùng header format như Network Data nhưng có payload chứa các lệnh điều khiển mạng cụ thể.

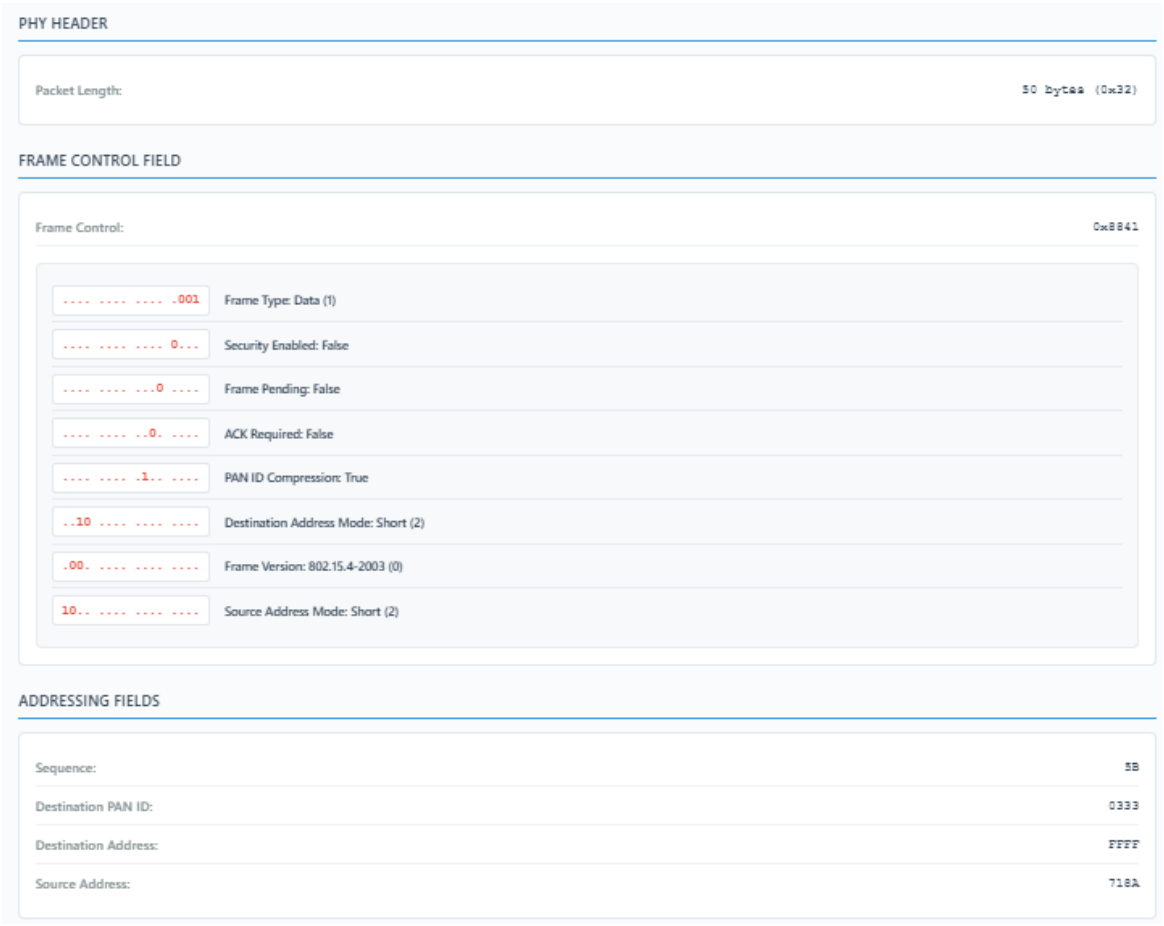
3.4.0.3 Tầng Zigbee security

Engine hỗ trợ đầy đủ 8 mức bảo mật ZigBee (0-7) với khả năng xử lý:

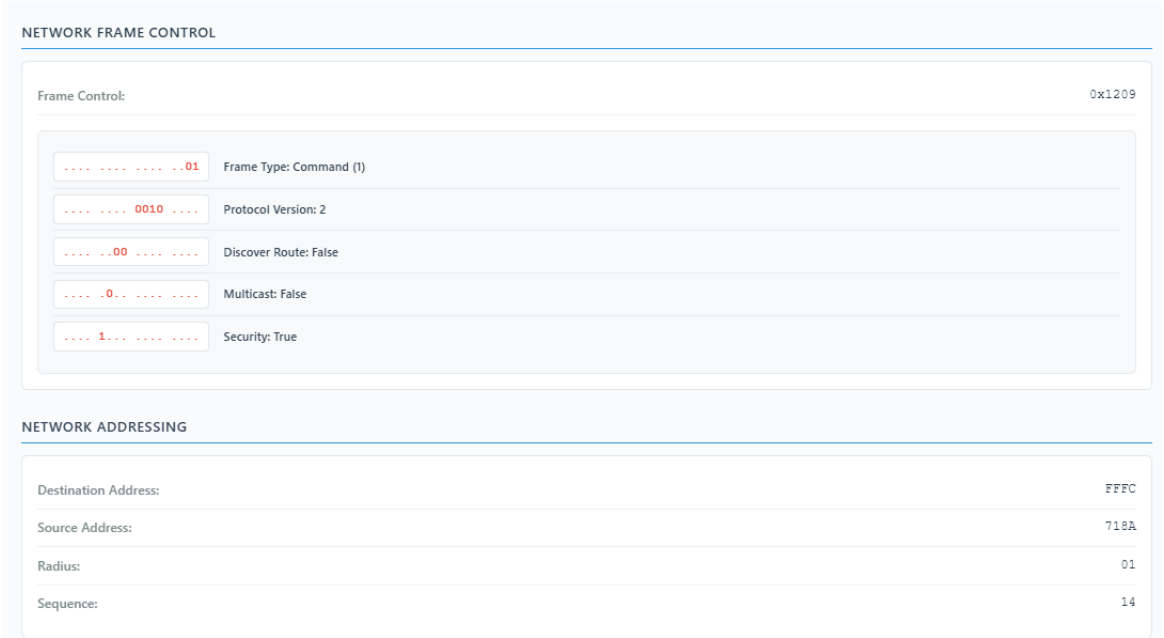
- Encryption Detection: Phân biệt giữa các gói tin có/không có mã hóa AES.
- MIC Validation: Xử lý Message Integrity Code với độ dài khác nhau (32/64/128-bit).
- Key Management: Hỗ trợ các loại key khác nhau (Network Key, Link Key, etc.).
- Nonce Handling: Xử lý Extended Nonce với 64-bit Source Address.
- Replay Protection: Sử dụng Frame Counter để chống tấn công lặp lại.

3.4.0.4 Kết quả giải mã các loại gói tin

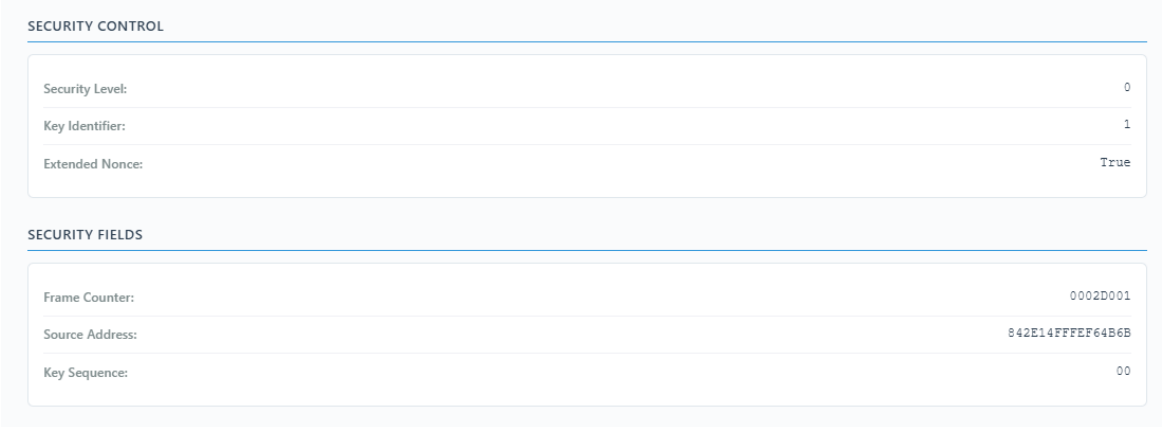
Zigbee Data là loại gói tin cơ bản dùng để truyền dữ liệu ứng dụng giữa các thiết bị trong mạng Zigbee. Các gói tin này có thể được gửi theo dạng unicast (từ một nguồn đến một đích cụ thể), multicast (từ một nguồn đến một nhóm thiết bị) hoặc broadcast (từ một nguồn đến tất cả các thiết bị trong mạng). Zigbee Data đóng vai trò trung tâm trong việc truyền tải thông tin cảm biến, điều khiển hoặc các dữ liệu ứng dụng khác giữa các thiết bị, đảm bảo mạng hoạt động đúng chức năng mong muốn. Hình 3.2 đến 3.5 là kết quả phân tích gói tin Zigbee data với raw data là "32 41 88 5B 33 03 FF FF 8A 71 09 12 FC FF 8A 71 01 14 6B 4B F6 FE FF 14 2E 84 28 01 D0 02 00 6B 4B F6 FE FF 14 2E 84 00 13 19 F9 93 06 F1 B5 D5 02"



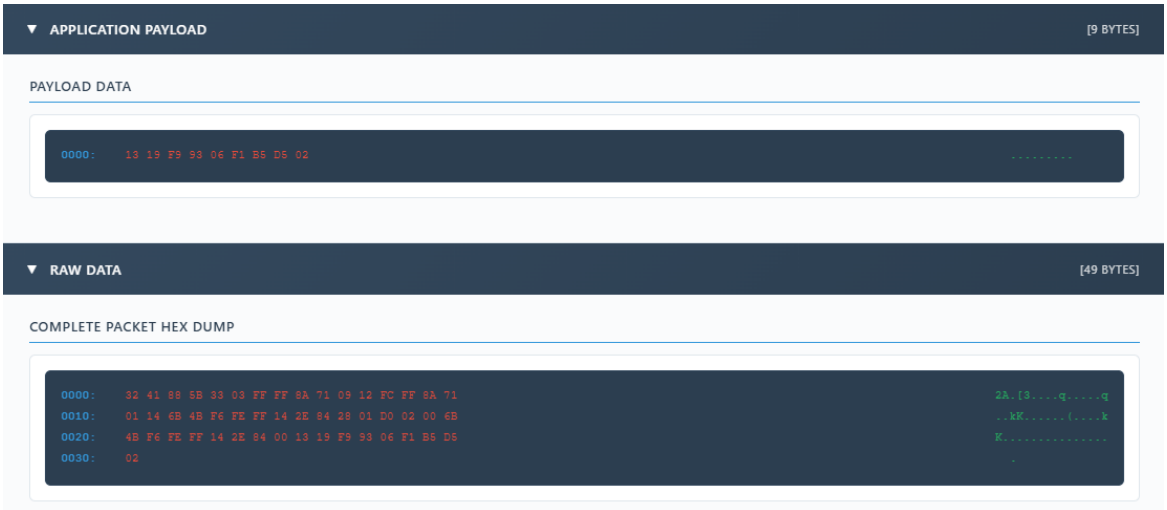
Hình 3.3. Kết quả giải mã tầng IEEE 802.15.4



Hình 3.4. Kết quả giải mã tầng Zigbee Network

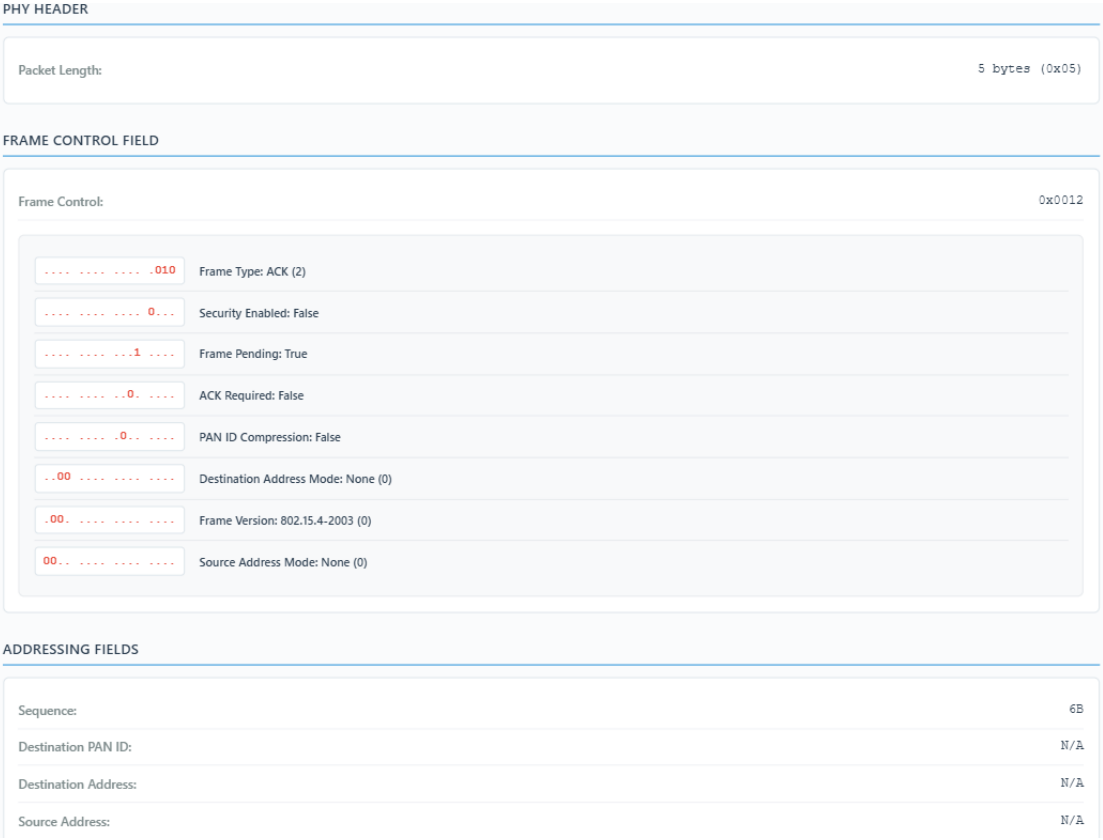


Hình 3.5. Kết quả giải mã tầng Zigbee Security



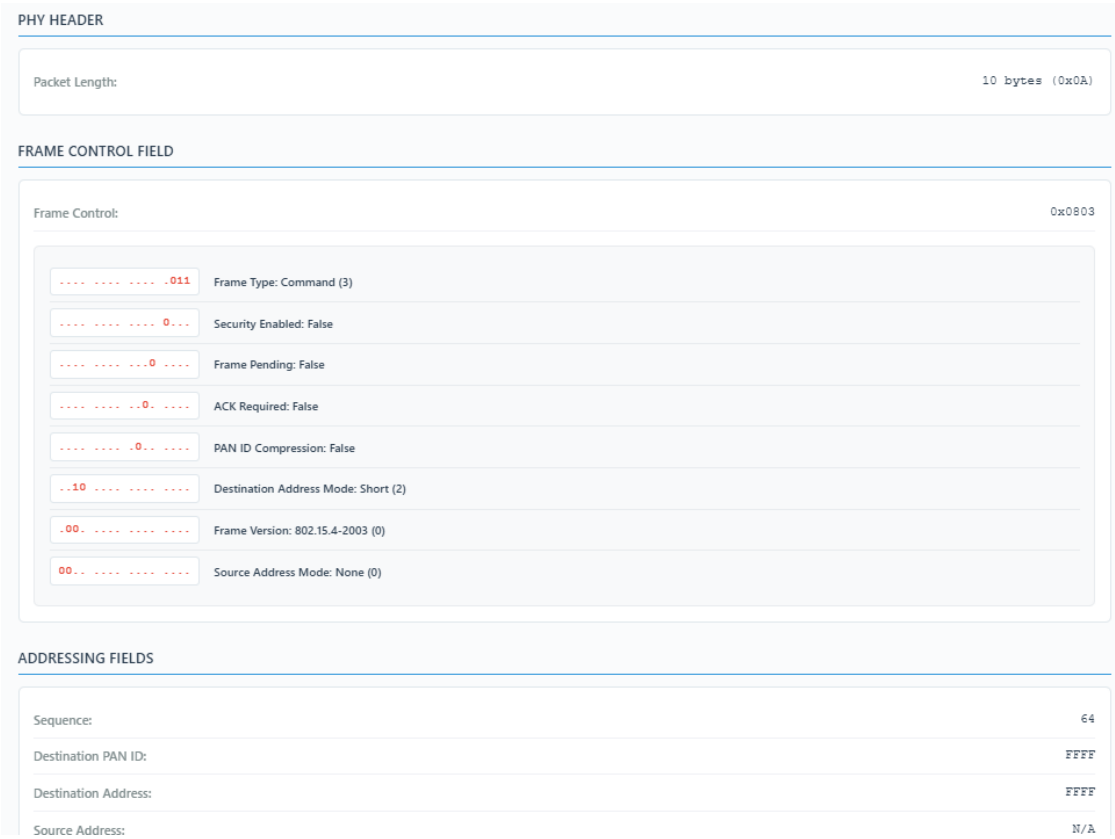
Hình 3.6. Kết quả giải mã Application Payload

Gói tin ACK là gói xác nhận được gửi từ thiết bị nhận về thiết bị gửi để báo rằng một gói dữ liệu đã được nhận thành công. Trong Zigbee, ACK tồn tại ở lớp MAC (theo chuẩn IEEE 802.15.4) và có thể được bật mặc định cho các gói unicast, giúp tăng độ tin cậy truyền thông bằng cách giảm khả năng mất gói tin. Nếu thiết bị gửi không nhận được ACK trong một khoảng thời gian nhất định, nó sẽ hiểu là gói tin bị lỗi và có thể gửi lại. Hình 3.6 là kết quả phân tích gói tin ACK với raw data là " 05 12 00 6B"



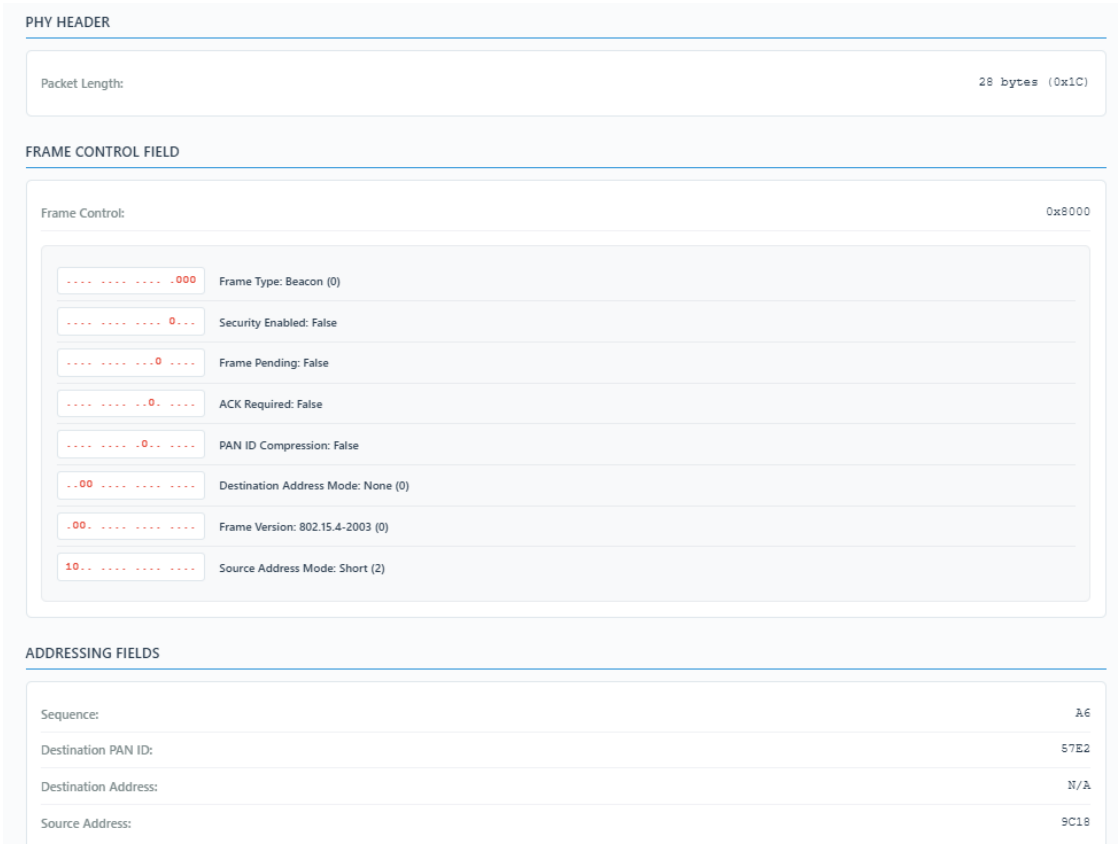
Hình 3.7. Kết quả giải mã gói ACK

Gói tin Beacon Request là gói broadcast do thiết bị mới (end device hoặc router) gửi ra khi muốn tìm kiếm các mạng Zigbee xung quanh. Khi nhận được Beacon Request, các coordinator hoặc router đang hoạt động trên kênh đó sẽ phản hồi bằng Beacon. Quá trình này giúp thiết bị mới phát hiện và lựa chọn mạng để gửi Association Request, đóng vai trò quan trọng trong quá trình khám phá và gia nhập mạng Zigbee. Hình 3.7 là kết quả phân tích gói tin Beacon Request với raw data là "0A 03 08 66 FF FF FF FF 07"



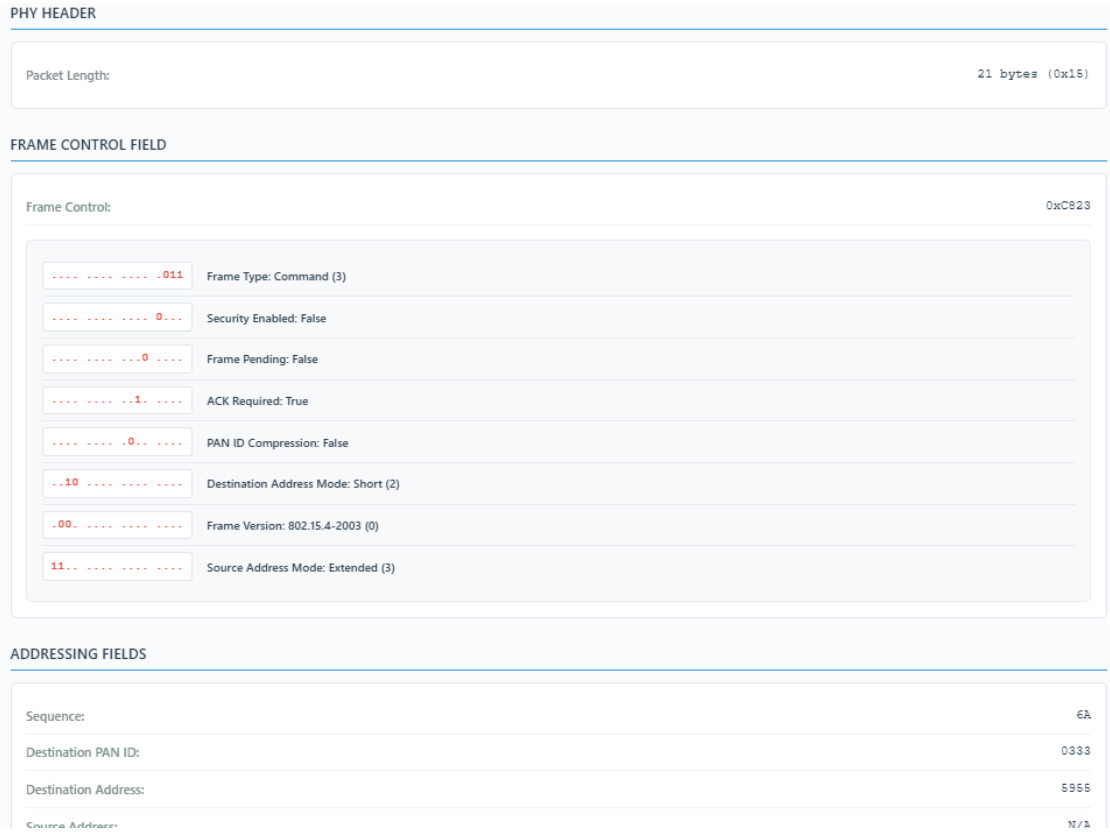
Hình 3.8. Kết quả giải mã gói tin Beacon Request

Gói tin Beacon là gói quảng bá được gửi định kỳ bởi coordinator hoặc router để thông báo sự tồn tại của mạng Zigbee trên một kênh tần số nhất định. Beacon chứa thông tin như PAN ID, thông số mạng, trạng thái cho phép thiết bị mới tham gia và các thông tin cấu hình khác. Các thiết bị mới quét kênh sẽ dựa vào Beacon để xác định mạng nào đang hoạt động và lựa chọn tham gia phù hợp. Hình 3.8 là kết quả phân tích gói tin Beacon với raw data là "1C 00 80 A6 E2 57 18 9C FF 0F 00 00 00 22 8C A6 4C FA F4 E3 A4 2A 3D FF FF FF 00"



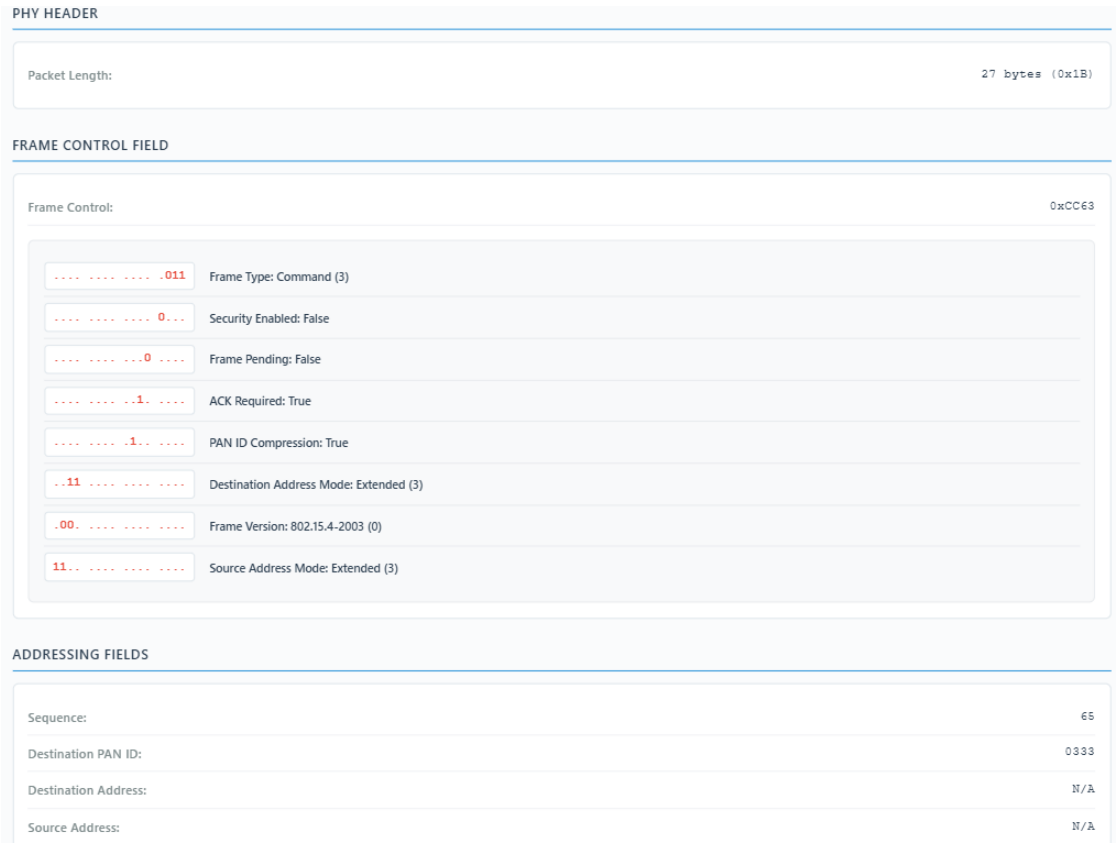
Hình 3.9. Kết quả giải mã gói tin Beacon

Gói tin Association Request được gửi từ thiết bị muốn tham gia mạng (thường là end device hoặc router) đến coordinator hoặc router cha. Gói tin này chứa thông tin nhận diện thiết bị và yêu cầu được phép gia nhập vào mạng Zigbee. Association Request là bước đầu tiên trong quá trình thiết bị mới đăng ký tham gia mạng, đóng vai trò khởi tạo mối quan hệ giữa thiết bị con và thiết bị cha trong cấu trúc mạng Zigbee. Hình 3.9 là kết quả phân tích gói tin Association Request với raw data là "15 23 C8 6A 33 03 55 59 FF FF 6B 4B F6 FE FF 14 2E 84 01 8E"



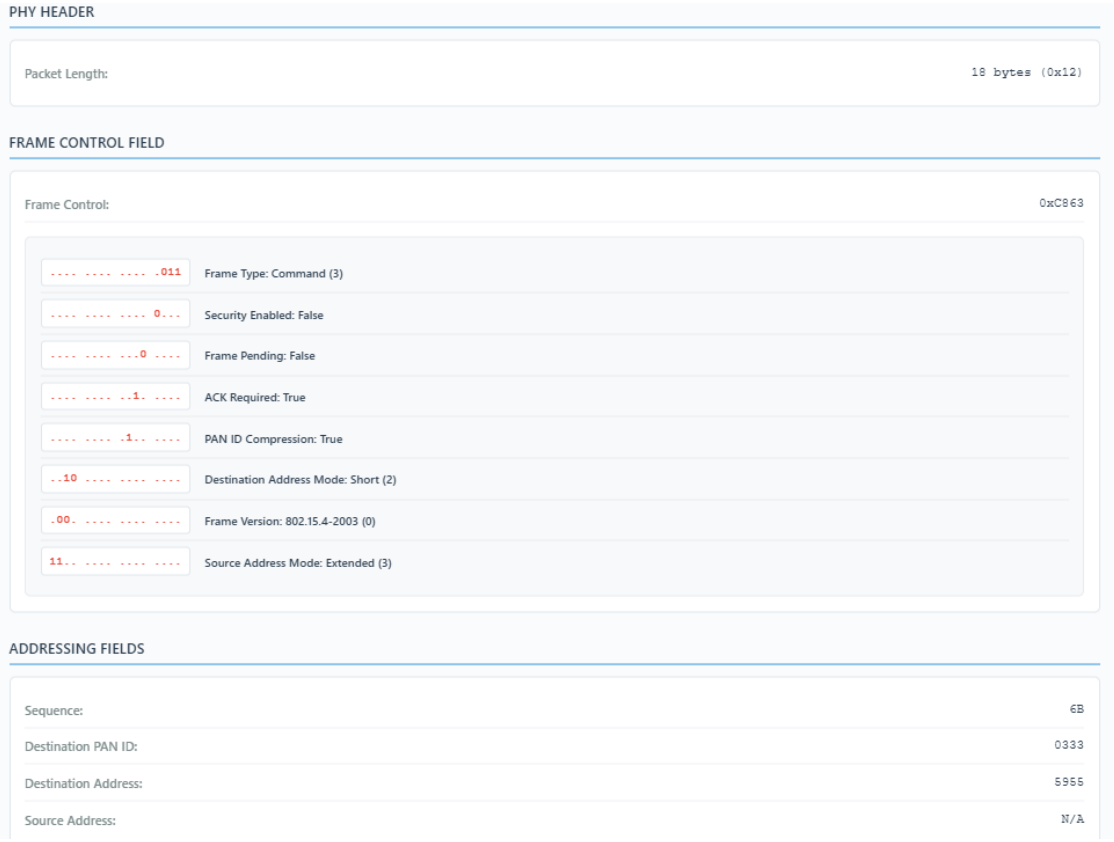
Hình 3.10. Kết quả giải mã gói tin Association Request

Sau khi nhận được Association Request, coordinator hoặc router sẽ gửi lại gói tin Association Response để trả lời cho thiết bị yêu cầu gia nhập. Association Response thông báo kết quả của quá trình đăng ký, bao gồm việc chấp nhận hoặc từ chối yêu cầu, cũng như cung cấp địa chỉ mạng (network address) cho thiết bị mới nếu được chấp nhận. Gói tin này hoàn tất quá trình kết nối thiết bị mới vào mạng Zigbee. Hình 3.10 là kết quả phân tích gói tin Association Response với raw data là "1B 63 CC 65 33 03 6B 4B F6 FE FF 14 2E 84 FE 4B F6 FE FF 14 2E 84 02 C6 7B 00"



Hình 3.11. Kết quả giải mã gói tin Association Response

Gói tin Data Request thường được gửi bởi end device đến thiết bị cha (coordinator hoặc router) để hỏi xem có dữ liệu nào đang chờ gửi cho nó hay không. Cơ chế này đặc biệt quan trọng với các thiết bị tiết kiệm năng lượng (sleepy end device), bởi chúng chỉ thức dậy định kỳ để gửi Data Request và nhận dữ liệu nếu có. Data Request giúp giảm tiêu thụ năng lượng và đảm bảo thiết bị vẫn nhận được thông tin cần thiết từ mạng. Hình 3.11 là kết quả phân tích gói tin Data Request với raw data là "12 63 C8 6B 33 03 55 59 6B 4B F6 FE FF 14 2E 84 04"



Hình 3.12. Kết quả giải mã gói tin Data Request

3.5 Xây dựng giao diện hiển thị

3.5.1 Các chức năng của giao diện Web

3.5.1.1 Quản lý node và kết nối phần cứng

Giao diện web cho phép người dùng trực tiếp thao tác với hệ thống phần cứng Zigbee một cách trực quan và thuận tiện nhất. Người dùng có thể:

- Tự động phát hiện và hiển thị các cổng COM khả dụng: Khi mở giao diện, người dùng có thể nhấn nút "Làm mới cổng" để hệ thống tự động quét các cổng serial đang kết nối với máy tính. Danh sách này luôn được cập nhật, giúp người dùng dễ dàng nhận biết các thiết bị mới vừa được cắm vào hoặc đã tháo ra.

- Thêm và khởi tạo node mới: Người dùng có thể chọn cổng COM, đặt tên node, cấu hình baudrate và nhấn "Kết nối" để tạo mới một node. Hệ thống sẽ tự động khởi tạo đối tượng node, mở kết nối UART, khởi động thread đọc dữ liệu và cập nhật trạng thái node lên giao diện.

- Theo dõi trạng thái kết nối real-time: Mỗi node được hiển thị dưới dạng một card riêng biệt trên dashboard, với các chỉ báo trạng thái như đang kết nối, đã kết nối, mất kết nối, hoặc đang thực hiện thao tác. Các trạng thái này được mã hóa màu sắc (xanh lá, vàng, đỏ, xanh dương) và có thể đi kèm hiệu ứng động để tăng tính trực quan.

- Đóng/mở kết nối node: Người dùng có thể đóng port, xóa node khỏi hệ thống hoặc khởi tạo lại node chỉ với một thao tác đơn giản trên giao diện mà không cần can thiệp thủ công vào phần cứng.

3.5.1.2 Điều khiển và giám sát thiết bị Zigbee

Giao diện web cung cấp bộ công cụ điều khiển mạnh mẽ cho từng node, giúp người dùng có thể thao tác từ xa với thiết bị Zigbee:

- Gửi lệnh điều khiển trực tiếp: Thông qua các nút chức năng, người dùng có thể gửi các lệnh như reset thiết bị, tạo mạng Zigbee mới, tham gia mạng, rời mạng, mở mạng cho phép thiết bị khác gia nhập, bật/tắt/toggle đèn LED trên node, hoặc cấu hình các tham số đặc biệt như SyncWords.
- Theo dõi trạng thái LED và các thông số thiết bị: Giao diện luôn cập nhật trạng thái LED của từng node (ON/OFF), đồng thời hiển thị các thông số như RSSI, LQI, kênh hoạt động, số lượng gói tin đã nhận, v.v. Mọi thay đổi trạng thái đều được phản ánh tức thời lên giao diện.
- Thực hiện các thao tác đặc thù Zigbee: Chẳng hạn, người dùng có thể yêu cầu node thực hiện Energy Scan, cấu hình lại thông số đồng bộ, hoặc gửi các lệnh đặc biệt phục vụ kiểm thử và gỡ lỗi.

3.5.1.3 Quản lý, hiển thị và phân tích gói tin realtime

- Bảng gói tin cập nhật liên tục: Tất cả các gói tin Zigbee nhận được từ các node sẽ được hiển thị trên một bảng dữ liệu lớn, với các trường thông tin như thời gian nhận, ID thiết bị, loại gói tin, giá trị RSSI, LQI, kênh, độ dài payload, trạng thái CRC, v.v. Bảng này có khả năng cập nhật real-time, đảm bảo người dùng luôn theo dõi được dữ liệu mới nhất.
- Phân loại và mã màu gói tin: Hệ thống tự động nhận diện loại gói tin (Beacon, Data, ACK, Command) dựa trên giải mã frame control field, và hiển thị chúng với màu sắc hoặc biểu tượng riêng biệt giúp người dùng dễ dàng nhận biết.
- Lọc và tìm kiếm nâng cao: Người dùng có thể lọc bảng theo loại gói tin, giá trị RSSI, kênh hoạt động, hoặc tìm kiếm nhanh theo ID thiết bị, hỗ trợ phân tích hiệu quả khi số lượng gói tin lớn.
- Xem chi tiết gói tin: Khi nhấn vào một gói tin bất kỳ, giao diện sẽ mở một modal hiển thị chi tiết từng trường của gói tin theo cấu trúc phân lớp (IEEE 802.15.4, Zigbee Network, Security, Payload), phân tích từng bit, từng trường, mô phỏng cách hiển thị của Wireshark. Người dùng có thể xem breakdown từng bit của frame control, các trường địa chỉ, thông tin bảo mật, và cả payload dưới dạng hexdump.

- Chức năng export dữ liệu: Người dùng có thể xuất dữ liệu gói tin ra các định dạng CSV, JSON hoặc PDF phục vụ cho việc phân tích ngoài hệ thống hoặc lưu trữ lâu dài.

3.5.1.4 Quản lý và hiển thị kết quả EScan

- Khởi động Energy Scan cho từng node: Người dùng có thể chọn node và bắt đầu quá trình quét năng lượng trên các kênh Zigbee chỉ với một thao tác trên giao diện.
- Hiển thị kết quả EScan trực quan: Kết quả quét năng lượng được cập nhật real-time và hiển thị dưới dạng bảng hoặc biểu đồ trực quan (bar chart/line chart), giúp người dùng nhận biết ngay các kênh bị nhiễu, từ đó chọn kênh tối ưu cho mạng Zigbee.
- Lưu trữ và truy xuất dữ liệu EScan: Toàn bộ kết quả quét được lưu trữ vào cơ sở dữ liệu, cho phép người dùng xem lại lịch sử quét, so sánh các lần quét khác nhau, hoặc xuất dữ liệu để phân tích sâu hơn.

3.5.1.5 Chức năng test BER

- Giao diện cấu hình kiểm thử BER: Cho phép người dùng chọn node phát (TX), node thu (RX), thiết lập các tham số kiểm thử như số bit, tần số, timeout, cấu hình SyncWords. Giao diện hỗ trợ kiểm tra trạng thái kết nối của từng node, cảnh báo khi cấu hình chưa hợp lệ.
- Theo dõi tiến trình kiểm thử BER real-time: Khi kiểm thử bắt đầu, giao diện hiển thị tiến trình kiểm thử với progress bar, số bit đã kiểm tra, số lỗi bit, tỷ lệ lỗi (BER), RSSI, cùng các chỉ báo trạng thái. Các thông số này được cập nhật liên tục, giúp người dùng theo dõi chất lượng đường truyền không dây trong thời gian thực.
- Dừng, xóa và reset kiểm thử dễ dàng: Giao diện hỗ trợ các nút dừng kiểm thử, xóa kết quả, reset node, đảm bảo quá trình kiểm thử luôn kiểm soát được và tránh các lỗi ngoài ý muốn.
- Hiển thị kết quả cuối cùng và lưu lịch sử: Sau khi kiểm thử hoàn thành, kết quả được lưu lại và hiển thị rõ ràng, có thể xem lại hoặc xuất báo cáo phục vụ cho việc đánh giá hiệu năng hệ thống.

3.6 Các API phục vụ người dùng và hướng dẫn triển khai Testbed

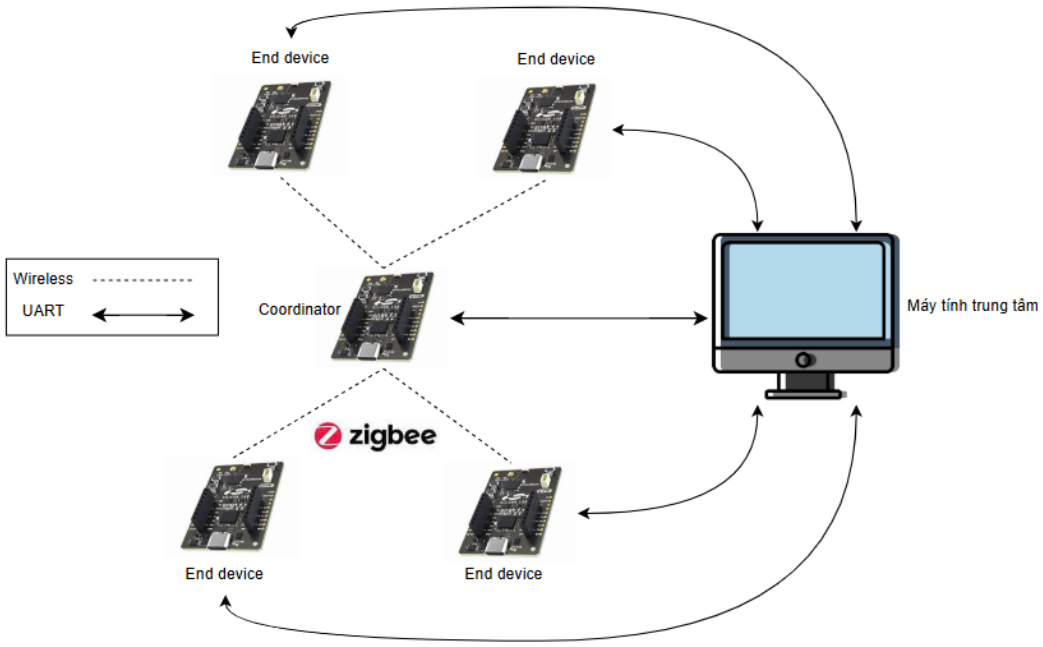
Kết luận chương

CHƯƠNG 4. KẾT QUẢ THỰC NGHIỆM

Mở đầu chương

4.7 Xây dựng kịch bản ứng dụng người dùng để kiểm thử hệ thống

4.7.1 Mô hình kịch bản ứng dụng

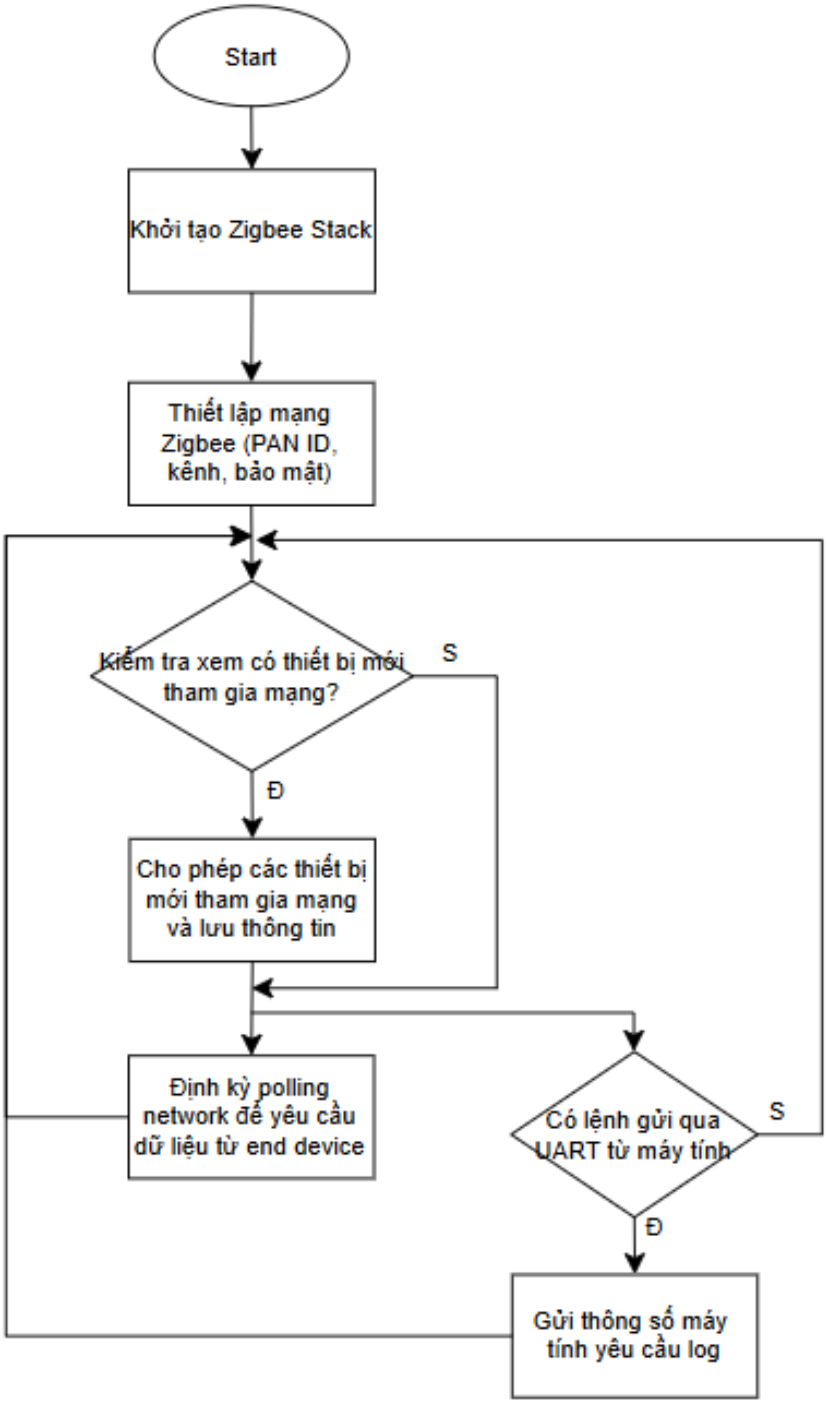


Hình 4.1. Sơ đồ mô hình kịch bản ứng dụng

Kịch bản ứng dụng người dùng được xây dựng là một mạng Zigbee hình sao gồm Coordinator ở trung tâm và 4 End device. Tất cả đều được kết nối UART đến máy tính trung tâm. Kết nối này là 2 chiều:

- Các thiết bị sẽ gửi dữ liệu các thông số lớp PHY và MAC về máy tính trung tâm để xử lý. Các thông số này gồm 2 loại: thông số tự động log và thông số log bởi người dùng khi cần
- Các thiết bị cũng sẽ nhận lệnh điều khiển từ máy tính trung tâm qua kết nối UART này

4.7.2 Xây dựng lưu đồ hoạt động cho các thành phần trong mạng



Hình 4.2. Lưu đồ hoạt động của Coordinator

Giai đoạn khởi tạo và thiết lập ban đầu: Coordinator bắt đầu quá trình hoạt động từ điểm start. Ngay sau đó, thiết bị tiến hành "Khởi tạo Zigbee Stack" - một bước quan trọng để thiết lập tất cả các lớp giao thức Zigbee cần thiết cho hoạt động mạng. Quá trình này bao gồm việc chuẩn bị các thành phần cốt lõi của stack protocol để Coordinator có thể đảm nhận vai trò trung tâm trong mạng.

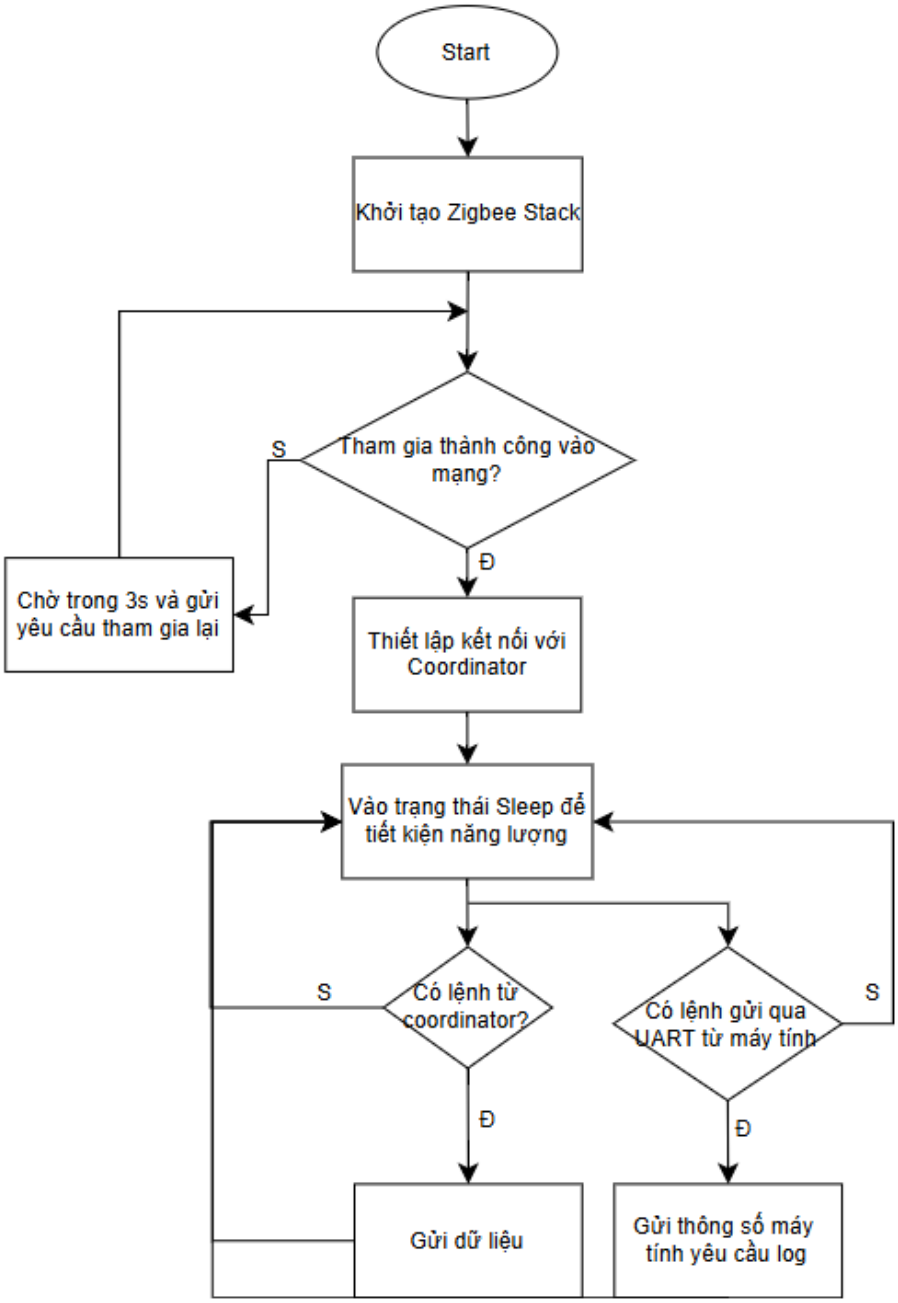
Quá trình thiết lập mạng Zigbee: Sau khi hoàn tất khởi tạo Stack, Coordinator

tiến hành "Thiết lập mạng Zigbee (PAN ID, kênh, bảo mật)". Đây là giai đoạn quan trọng nhất trong việc tạo lập mạng, bao gồm việc chọn PAN ID duy nhất, lựa chọn kênh hoạt động phù hợp và thiết lập các chính sách bảo mật cho toàn bộ mạng. Coordinator đóng vai trò là người khởi tạo và quản lý tất cả các thông số mạng này.

Cơ chế quản lý thiết bị tham gia mạng: Sau khi thiết lập mạng hoàn tất, Coordinator kiểm tra xem có thiết bị nào tham gia mạng?". Nếu không có thiết bị mới, hệ thống tạo thành một vòng lặp để tiếp tục giám sát. Ngược lại, Coordinator sẽ thực hiện "Cho phép các thiết bị mới tham gia mạng và lưu thông tin". Quá trình này đảm bảo Coordinator có thể tiếp nhận và quản lý các End device mới một cách liên tục.

Hoạt động polling và quản lý dữ liệu: Một khi đã có thiết bị tham gia mạng, Coordinator chuyển sang chế độ hoạt động chính với việc "Định kỳ polling network để yêu cầu dữ liệu từ end device". Đây là cơ chế quan trọng trong mạng Zigbee, cho phép Coordinator chủ động thu thập dữ liệu từ các End device, đặc biệt là những thiết bị hoạt động ở chế độ tiết kiệm năng lượng và không thể gửi dữ liệu một cách chủ động.

Giao tiếp với máy tính thông qua UART: Đồng thời với hoạt động polling, Coordinator cũng kiểm tra xem có lệnh gửi qua UART từ máy tính. Nếu không có lệnh, hệ thống quay lại chu trình polling network. Ngược lại, Coordinator sẽ thực hiện "Gửi thông số máy tính yêu cầu log", cho phép giao tiếp và truyền dữ liệu giữa mạng Zigbee và hệ thống máy tính trung tâm.



Hình 4.3. Lưu đồ hoạt động của End device

Giai đoạn khởi động và thiết lập ban đầu: Khi end device được khởi động, thiết bị sẽ bắt đầu từ trạng thái Start và ngay lập tức tiến hành khởi tạo Zigbee Stack. Đây là bước quan trọng để thiết lập tất cả các lớp giao thức cần thiết cho hoạt động Zigbee, bao gồm các lớp vật lý, MAC, mạng và ứng dụng.

Quá trình gia nhập mạng: Sau khi khởi tạo Zigbee Stack hoàn tất, end device sẽ kiểm tra xem có tham gia thành công vào mạng hay không. Nếu quá trình tham gia mạng chưa thành công, thiết bị sẽ chờ trong vòng 3 giây và sau đó gửi yêu cầu tham gia lại. Chu trình này sẽ được lặp lại liên tục cho đến khi thiết bị có thể tham gia thành công vào mạng Zigbee.

Thiết lập kết nối với Coordinator: Một khi end device đã tham gia thành công

vào mạng, bước tiếp theo là thiết lập kết nối trực tiếp với Coordinator. Đây là mối quan hệ cha-con quan trọng trong cấu trúc mạng Zigbee, cho phép End device giao tiếp và trao đổi dữ liệu thông qua Coordinator.

Chế độ tiết kiệm năng lượng: Sau khi hoàn tất quá trình thiết lập, End device sẽ chuyển vào trạng thái sleep để tiết kiệm năng lượng. Đây là trạng thái mặc định của thiết bị nhằm tối ưu hóa tuổi thọ pin, đặc biệt quan trọng đối với các thiết bị hoạt động bằng pin trong thời gian dài.

Cơ chế xử lý sự kiện song song: Từ trạng thái sleep, End device sẽ liên tục theo dõi hai loại sự kiện khác nhau một cách song song. Thứ nhất, thiết bị kiểm tra xem có lệnh nào từ Coordinator hay không. Nếu có lệnh từ coordinator, end device sẽ thực hiện việc gửi dữ liệu theo yêu cầu, sau đó quay lại trạng thái sleep. Đồng thời, End device cũng kiểm tra xem có lệnh nào được gửi qua UART từ máy tính hay không. Khi phát hiện có lệnh từ máy tính, thiết bị sẽ thực hiện việc gửi thông số mà máy tính yêu cầu log. Sau khi hoàn thành nhiệm vụ này, thiết bị cũng sẽ quay trại trạng thái sleep để tiết kiệm năng lượng.

4.8 Thử nghiệm hệ thống trong các điều kiện khác nhau

4.8.1 Môi trường ít nhiễu

4.8.2 Môi trường nhiều nhiễu tín hiệu

4.9 Tính toán PER và PLR từ các event trong database

4.9.1 Công thức tính toán PER

4.9.2 Công thức tính toán PLR

4.9.3 Kết quả

Kết luận chương

KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

TÀI LIỆU THAM KHẢO

PHỤ LỤC

A Một số phương pháp đo và hiệu chuẩn

Phụ lục cần thêm (nếu có) ...